대 한 민 국 특 허 청
KOREAN INTELLECTUAL PROPERTY OFFICE

별첨 사본은 아래 출원의 원본과 동일함을 증명함.

This is to certify that the following application annexed hereto is a true copy from the records of the Korean Intellectual Property Office.

출 원 번 호 : 10-2002-0062923
Application Number

출 원 년 월 일 : 2002년 10월 15일
Date of Application OCT 15, 2002

출 원 인 : 삼성전자주식회사
Applicant(s) SAMSUNG ELECTRONICS CO., LTD.

2003 년 07 월 07 일

특 허 청
COMMISSIONER

【서지사항】

【서류명】	특허출원서
【권리구분】	특허
【수신처】	특허청장
【참조번호】	0002
【제출일자】	2002.10.15
【발명의 명칭】	디지털 컨텐츠 메타데이터의 복합 조건 검색을 위한 멀티 키 인덱스 정보 스트림 구조와 디지털 컨텐츠 메타데이터 검색 방법 및 장치
【발명의 영문명칭】	MULTI-KEY INDEX DATA STREAM STRUCTURE, METHOD AND APPARATUS FOR MULTI-KEY SEARCH FOR METADATA OF DIGITAL CONTENTS USING THIS STRUCTURE
【출원인】	
【명칭】	삼성전자 주식회사
【출원인코드】	1-1998-104271-3
【대리인】	
【성명】	김동진
【대리인코드】	9-1999-000041-4
【포괄위임등록번호】	2002-007585-8
【발명자】	
【성명의 국문표기】	신효섭
【성명의 영문표기】	SHIN,Hyo Seop
【주민등록번호】	711013-1528617
【우편번호】	157-203
【주소】	서울특별시 강서구 가양3동 6단지 아파트 612동 1310호
【국적】	KR
【우선권주장】	
【출원국명】	KR
【출원종류】	특허
【출원번호】	10-2002-0043097
【출원일자】	2002.07.23
【증명서류】	미첨부
【심사청구】	청구

1020020062923 출력 일자: 2003/7/8

【취지】 특허법 제42조의 규정에 의한 출원, 특허법 제60조의 규정에 의한 출원심사 를 청구합니다. 대리인
김동진 (인)

【수수료】

【기본출원료】	20	면	29,000 원
【가산출원료】	32	면	32,000 원
【우선권주장료】	1	건	26,000 원
【심사청구료】	6	항	301,000 원
【합계】	388,000 원		

【첨부서류】 1. 요약서·명세서(도면)_1통 2.우선권증명서류 및 동 번역문[특허청 기제출]_1통

【요약서】

【요약】

본 발명은 디지털 컨텐츠에 관한 메타데이터, 특히 TV-Anytime 포럼에서 정의하고 있는 디지털 컨텐츠 메타데이터(TVA 메타데이터)에 대한 디지털 컨텐츠 메타데이터의 복합 조건 검색을 위한 인덱스 정보 스트림 구조와 디지털 컨텐츠 메타데이터 검색 방법 및 수신 장치에 관한 것으로, 디지털 컨텐츠에 관한 메타데이터의 멀티키 인덱싱이 가능한 인덱스 정보 스트림 구조를 제공함으로써 디지털 컨텐츠에 관한 메타데이터를 수신하는 수신장치가 다수의 조건 즉, 복합 조건을 충족하는 메타데이터를 보다 신속하고 효율적으로 검색할 수 있도록 하는 인덱스 정보 스트림 구조와 디지털 컨텐츠 메타데이터 검색 방법 및 수신장치에 관한 것이다.

본 발명은 TVA 메타데이터가 프래그먼트 단위로 전송되는 전송 스트림 환경에서, TVA 메타데이터 프래그먼트에 대한 보다 효율적인 검색 및 접근 기능을 제공하기 위해 멀티키 인덱싱 접근 방법을 제시함으로써 TVA 메타데이터를 수신하는 수신장치가 TVA 메타데이터들에 대한 복합 조건 검색을 보다 효율적으로 수행할 수 있게 된다.

【대표도】

도 9

【색인어】

TV-Anytime, 메타데이터, 프래그먼트, 멀티키, 인덱스

【명세서】

【발명의 명칭】

디지털 컨텐츠 메타데이터의 복합 조건 검색을 위한 멀티키 인덱스 정보 스트림 구조와 디지털 컨텐츠 메타데이터 검색 방법 및 장치{MULTI-KEY INDEX DATA STREAM STRUCTURE, METHOD AND APPARATUS FOR MULTI-KEY SEARCH FOR METADATA OF DIGITAL CONTENTS USING THIS STRUCTURE}

【도면의 간단한 설명】

도 1은 일반적인 PDR의 개념도이다.

도 2는 일반적인 EPG 애플리케이션에서의 그리드 가이드 화면을 도시한다.

도 3은 TV-Anytime 포럼에서 정의하는 일반적인 메타데이터의 구조를 도시한다.

도 4는 TV-Anytime 포럼에서 정의하는 일반적인 프래그먼트의 개념도이다.

도 5는 TV-Anytime 포럼에서 정의하는 일반적인 컨테이너 개념도이다.

도 6은 종래의 싱글키 개념의 인덱스 정보 스트림 구조를 도시한다.

도 7은 종래의 싱글키 개념을 이용한 인덱스 정보의 자료 구조 및 검색과정을 도시한다.

도 8은 종래의 싱글키를 이용한 인덱스 정보 스트림에 대한 검색방법을 설명한 도면이다.

도 9는 본 발명에 따른 멀티키 개념의 인덱스 정보 스트림 구조를 도시한다.

도 10은 본 발명에 따른 멀티키 개념을 이용한 인덱스 정보의 자료 구조 및 검색과정을 도시한다.

도 11은 본 발명에 따른 멀티키를 이용한 인덱스 정보 스트림에 대한 검색방법을 도시한다.

도 12는 본 발명에 따른 멀티키 인덱스 정보 스트림을 수신하는 수신장치의 구조도이다.

도 13은 본 발명에 따른 멀티키 인덱스 정보 스트림을 수신하는 수신장치의 동작 흐름도이다.

【발명의 상세한 설명】

【발명의 목적】

【발명이 속하는 기술분야 및 그 분야의 종래기술】

<14> 본 발명은 디지털 컨텐츠 메타데이터에 대한 복합 조건 검색을 위한 멀티키 인덱스 정보 스트림 구조와 이를 이용한 메타데이터 검색방법, 그리고 상기 인덱스 정보 스트림을 수신하는 수신장치에 관한 것으로서, 특히 TV-Anytime 포럼에서 정의하고 있는 디지털 컨텐츠에 대한 XML 메타데이터(이하 'TVA 메타데이터')를 프래그먼트라는 독립적인 단위로 나누어 전송할 때, 메타데이터에 대한 복합 조건 검색을 보다 효율적으로 수행할 수 있도록 TVA 메타데이터 프래그먼트에 대한 멀티키 인덱스를 포함하는 인덱스 정보 스트림 구조와 이를 이용한 TVA 메타데이터 검색방법 및 수신장치에 관한 것이다.

<15> TV-Anytime 포럼은, 개인용 대용량 저장장치를 가진 PDR(Personal Digital Recorder)과 같은 사용자 환경에서 오디오 비주얼 관련 서비스 제공을 위한 표준

개발을 목적으로 1999년 9월에 설립된 민간 표준 기구로서, 모든 사용자가 개인용 저장 장치를 기반으로 자기가 원하는 방법으로 원하는 시간에 다양한 형태(기존의 방송 서비스 및 온라인 대화형 서비스 등)의 프로그램을 시청할 수 있게 하는 것을 그 구체적인 서비스 목표로 하고 있다.

<16> TV-Anytime 포럼은 비즈니스 모델, 시스템/전송 인터페이스/컨텐츠 참조, 기술, 메타데이터, 컨텐츠 보호관리 등의 워킹 그룹으로 나누어 표준화를 진행하고 있으며, 본 발명과 관련된 메타데이터에 대해서는 2002년 6월 현재 "1st Draft of Metadata Specification SP003v1.3"까지 발표되었다.

<17> 도 1을 참조하여 PDR의 구성을 간략하게 살펴보면, PDR(100)은 영상/음성 신호를 제공하는 제공자(200)로부터 공중파, 위성파, 인터넷망 등의 다양한 네트웍을 통해 영상/음성 신호 및 메타데이터를 수신하고, 필요에 따라 시청자의 시청 패턴, 기호 등을 수집하여 영상/음성 신호를 제공하는 제공자(200)에게 송신한다. PDR(100)은 수신된 영상/음성 신호 및 메타데이터를 저장하는 대용량 저장장치, 영상/음성 신호의 저장 및 재생을 위한 소프트웨어 및 영상/음성 신호에 대한 메타데이터를 검색/전시하기 위한 EPG(Electronic Program Guide) 애플리케이션을 포함하고, 사용자는 도 2에 도시된 EPG 애플리케이션의 그리드 가이드 화면 등을 통해 영상/음성 데이터에 관한 메타데이터 즉, 프로그램 제목, 프로그램 재생 시간 등을 확인하고 원하는 프로그램을 선택하여 네트웍을 통해 실시간으로 수신하거나 또는 이미 대량 저장장치에 저장된 영상/음성 데이터를 재생시킨다.

<18> 메타데이터는 프로그램 타이틀과 개요와 같은 컨텐츠에 대한 설명적인 데이터를 의미하며, "데이터에 관한 데이터(data about data)"로 정의된다. TV-Anytime 포럼의 TVA 메타데이터 규격에서는 XML에 관한 표준기구 W3C의 표준인 XML Schema 언어(W3C의 XML 1.0 참조)를 사용하여 그 구조를 정의하는 한편, 각 메타데이터 엘리먼트 및 속성에 대한 의미(semantics)를 함께 규정하고 있다. 방송 컨텐츠에 관련된 TVA 메타데이터는 도 3에 도시된 바와 같이 "TVAMain" (300)이라는 루트 노드를 가지는 XML 문서로 구성되며, 예를 들어 프로그램에 관한 메타데이터들은 "ProgramDescription" 노드 이하에서 프로그램 정보(ProgramInformation Table), 그룹 정보(GroupInformation Table), 프로그램 위치 정보(ProgramLocation Table), 서비스 정보(ServiceInfomation Table) 등의 노드들로 구성된다.

<19> TV-Anytime 포럼은 방대한 TVA 메타데이터를 스트림으로 전송하기 위해 프래그먼트(fragment)라는 독립적인 단위로 TVA 메타데이터를 전송한다. 도 4를 참조하여 프래그먼트의 개념을 간략히 설명하면, 프래그먼트란 도 3에 도시된 XML 문서로 구성된 TVA 메타데이터를 소정의 트리 구조별로 분할한 것으로서, 예를 들면, 전체 TVA 메타데이터 를 "TVAMain" 을 상위 노드로 하여 소정의 자식 노드까지 포함하는 트리 구조 (프래그먼트 TVAMain), 프로그램 정보(ProgramInformation Table)를 상위 노드로 하여 이하 자식 노드까지 포함하는 트리 구조(프래그먼트 ProgramInformation), 브로드캐스트이벤트 정보(BroadcastEvent)를 상위 노드로 하여 이하 자식 노드까지 포함하는 트리 구조(프래그먼트 BroadcastEvent)등으로 분할할 경우, 분할된 트리 구조 각각은 프래그먼트가 되고, 이 프래그먼트들은 다른 프래그먼트와 무관하게 독립적으로 전송가능하고 또한 개별 접근이 가능하다.

<20> 프래그먼트들에 대한 개별 접근을 위하여, 전송된 TVA 메타데이터 프래그먼트가 전체 메타데이터 트리 구조에서 어떤 노드를 참조하는지 즉, TVA 메타데이터 프래그먼트의 상위 노드가 어떤 노드에 해당하는지, 그리고 전송된 TVA 메타데이터 프래그먼트내에 포함된 키('키' 는 프래그먼트가 참조하는 노드의 자식 노드을 의미하며, 사용자가 입력하게 되는 정보 필드(검색 조건), 예를 들면 '채널번호' 또는 '방송시간' 등이 키에 해당한다)들의 상기 TVA 메타데이터 프래그먼트내 상대 경로를 표시할 필요가 있으며, 이를 위해 W3C에서 정의한 XPath(XPath는 W3C에서 정의한 XML 문서내 노드에 대한 경로를 기술하는 신텍스(syntax)임)를 사용한다.

<21> 한편, 프래그먼트들에 대한 효율적인 검색과 접근을 제공하기 위해서는 메타데이터 프래그먼트에 포함된 키에 대한 인덱스 구조가 별도로 필요하고, 이런 인덱스 구조에 관한 정보 즉, 인덱스 정보 또한 상기 메타데이터 프래그먼트들과는 독립적으로 전송된다.

<22> TV-Anytime 포럼이 제공하는 환경하에서, 소정 방송시간 조건에 해당하는 프로그램 정보를 사용자가 검색하고자 할 경우, 먼저 프래그먼트들과는 독립적으로 전송되는 인덱스 정보를 활용하여 원하는 방송시간 조건에 해당하는 메타데이터 프래그먼트의 위치(식별자)를 파악하고, 이 위치(식별자)를 기초로 해당 메타데이터 프래그먼트에 접근하여 상기 방송시간 조건에 해당하는 메타데이터를 추출하게 된다.

<23> 종래기술문헌(TV-Anytime Specification TV145, J.P. Evain, "1st Draft of Metadata Specification SP003v1.3," TV-Anytime 포럼 17차 회의, 몬트리올, 캐나다, 2002년 6월, 이하 '싱글키 인덱스 기술문헌')에서는 메타데이터 프래그먼트 인덱스를 위한 싱글키 인덱스 정보 스트림 구조를 제안하고 있다.

<24> 참고로 싱글키는 이후 설명되는 본 발명의 멀티키에 구별하기 위해 본 명세서에서 사용하는 개념으로, 본 발명의 멀티키 인덱스 정보 스트림 구조는 동시에 다수의 키를 사용하여 다수의 키에 해당하는 메타데이터를 접근할 수 있으나, 종래기술의 싱글키 인덱스 정보 스트림 구조는 메타데이터 접근시 오직 하나의 키만을 사용할 수 있다.

<25> 인덱스 정보 스트림 구조를 설명하기에 앞서 TV-Anytime 포럼이 정의하는 컨테이너 개념을 먼저 설명하기로 한다.

<26> TV-Anytime 포럼은 앞서 설명한 인덱스 정보와 메타데이터 프래그먼트 등의 모든 데이터가 전송되는 최상위 저장소 즉, 컨테이너(container)라는 최상위 전송 형태를 정의하는데, 컨테이너를 개략적으로 설명하면, 모든 컨테이너는 다수의 섹션들로 구성되고, 각 섹션에 상기 인덱스 정보와 메타데이터 프래그먼트들이 저장된다. 컨테이너는 운반하는 정보에 따라 키 인덱스 리스트(key_index_list) 섹션, 키 인덱스(key_index) 섹션, 서브키 인덱스(sub_key_index) 섹션, 스트링 저장소(string_repository) 섹션, 프래그먼트 데이터 저장소(fragment_data_repository) 섹션 등의 인덱스 정보 섹션을 운반하는 인덱스 컨테이너와 엘리먼트 테이블(elements_table) 섹션, 스트링 저장소(string_repository) 섹션, 프래그먼트 저장소(fragment_data_repository) 섹션 등의 메타데이터 프래그먼트 섹션을 운반하는 데이터 컨테이너로 구분된다. 위 구분은 컨테이너에 포함된 정보의 내용에 따라 구분하는 것으로서, 컨테이너의 구성은 인덱스 컨테이너와 데이터 컨테이너 모두 동일하다.

<27> 도 5을 참조하여 TV-Anytime 포럼에서 정의한 컨테이너를 살펴보면, 컨테이너는 미도시된 컨테이너 식별자 정보(container_id) 필드와 다수의 섹션들을 포함하며, 각 섹션들은 'section_id' 의 인코딩 값에 따라 ' section_body' 에 저장된 내용이 식별된다.

예를 들면, 'section_id' 의 인코딩 값이 '0X0004' 인 섹션(10)은 키 인덱스 리스트(key_index_list) 섹션, 인코딩 값이 '0X0005' 인 섹션(20)은 키 인덱스(key-index) 섹션, 인코딩 값이 '0X0006' 인 섹션(30)은 서브키 인덱스(sub_key_index) 섹션, 인코딩 값이 '0X0001' 인 섹션(40)은 엘리먼트 테이블(element_table) 섹션, 인코딩 값이 '0X0003' 인 섹션(50)은 프래그먼트 데이터 저장소(fragment_data_repository) 섹션으로 각각 식별된다.

<28> TVA 메타데이터 프래그먼트는 데이터 컨테이너의 프래그먼트 데이터 저장소(fragment_data_repository) 섹션(50)내에 저장되어 전송되는데, 데이터 컨테이너는 상기 데이터 컨테이너 내의 TVA 메타데이터 프래그먼트들에 대한 식별자 정보(handle_value) 필드를 엘리먼트 테이블 섹션(40)에 포함한다.

<29> 결론적으로, TVA 메타데이터 프래그먼트는 TVA 메타데이터 프래그먼트가 포함된 컨테이너의 컨테이너 식별자 정보(container_id)와 메타데이터 프래그먼트 식별자 정보(handle_value)에 의해 유일하게 식별된다.

<30> 싱글키 인덱스 기술문헌에서는 전술한 데이터 컨테이너에 저장된 TVA 메타데이터 프래그먼트를 인덱스하기 위한 싱글키 인덱스 정보 스트림 구조 즉, 키 인덱스 리스트(key_index_list) 섹션(10), 키 인덱스(key_index) 섹션(20), 및 서브키 인덱스(sub_key_index) 섹션(30)의 구조를 제안하고 있다. 상기 구조의 신텍스(syntax)에 대해서는 싱글키 인덱스 기술문헌에 자세히 기술되어 있으므로 이에 대한 구체적인 설명은 생략하기로 하고, 이하에서는 상기 구조를 인덱스 정보 스트림상의 세그먼트로 표현한 도 6을 참조하여 상기 구조를 살펴보기로 한다.

<31> 싱글키 인덱스 정보 스트림 구조에서 정의되는 키 인덱스 리스트(key_index_list) 섹션(10)은, 전송되는 모든 싱글키의 리스트를 제공한다. 이 리스트에는 각 싱글키들에 대한 싱글키 정보(즉, 싱글키가 포함된 메타데이터 프래그먼트에 대한 XPath(fragment_xpath_ptr)와, 싱글키로서 사용되는 노드의 상기 프래그먼트의 XPath 위치에서의 관련 경로 즉, 싱글키로 사용되는 노드들의 해당 프래그먼트내 상대 경로에 대한 XPath(key_xpath_ptr))와, 후술되는 키 인덱스(key_index) 섹션(20)에 대한 식별 정보가 포함된다.

<32> 상기 메타데이터 프래그먼트의 XPath는 TVA 메타데이터 XML 문서의 루트 노드에 대한 경로 즉, 절대 경로이며, 싱글키로 사용되는 노드들의 XPath 즉, 싱글키의 XPath는 상기 메타데이터 프래그먼트에 대한 싱글키의 상대 경로를 나타낸다. 메타데이터 프래그먼트에 대한 XPath와 싱글키에 대한 XPath는 각각 'fragment_xpath_ptr' 세그먼트(11)와 'key_xpath_ptr' 세그먼트(12)에 저장된다.

<33> 또한, 키 인덱스 리스트(key_index_list) 섹션(10)에는 후술되는 각 싱글키의 키 인덱스(key_index) 섹션(20)에 대한 식별 정보(즉, 키 인덱스(key_index) 섹션(20)이 저장된 컨테이너의 컨테이너 식별자 정보(container_id) 및 키 인덱스 식별자 정보)가 포함되는데, 상기 컨테이너 식별자 정보 및 키 인덱스 식별자 정보들은 각각 키 인덱스 리스트(key_index_list) 섹션(10)의 'index_container' 세그먼트 및 'key_index_identifier' 세그먼트에 저장되어 전송된다.

<34> 싱글키 인덱스 정보 스트림 구조에서 정의되는 키 인덱스(key_index) 섹션(20)은, 후술되는 모든 서브키 인덱스(sub_key_index) 섹션(30)의 리스트를 제공하는데, 이 리스트에는 각 서브키 인덱스(sub_key_index) 섹션(30)들에 포함된 키값들의 범위를 나타내

는 정보 즉, 각 서브키 인덱스(sub_key_index) 섹션(30)내 키값들 중 가장 큰 키값(이하 '대표키 값'), 및 각 대표키 값에 관련된 서브키 인덱스(sub_key_index) 섹션(30)에 대한 식별 정보(즉, 서브키 인덱스(sub_key_index) 섹션이 저장된 컨테이너의 컨테이너 식별자 정보(container_id) 및 서브키 인덱스 식별자 정보)가 포함된다.

<35> 키 인덱스(key_index) 섹션(20)은 키 인덱스 리스트(key_index_list) 섹션(10)에서 정의된 키 인덱스 식별자 정보가 저장되는 'key_index_identifier' 세그먼트, 각 서브키 인덱스(sub_key_index) 섹션(30)의 대표키 값이 저장된 'high_key_value' 세그먼트(13), 대표키 값이 나타내는 범위에 해당하는 키값들을 포함하는 서브키 인덱스(sub_key_index) 섹션(30)에 대한 식별 정보로서, 서브키 인덱스(sub_key_index) 섹션(30)이 저장된 컨테이너의 컨테이너 식별자 정보(container_id)와 서브키 인덱스 식별자 정보가 각각 저장된 'sub_index_container' 세그먼트 및 'sub_index_identifier' 세그먼트들을 포함한다.

<36> 싱글키 인덱스 정보 스트림 구조에서 정의되는 서브키 인덱스(sub_key_index) 섹션(30)은, 해당 서브키 인덱스(sub_key_index) 섹션(30)에 포함된 키값들에 대한 리스트를 제공하는데, 이 리스트에는 해당 서브키 인덱스(sub_key_index) 섹션(30)에 포함된 키값들 및 상기 키값들을 가지는 메타데이터 프래그먼트에 대한 식별 정보(즉, 메타데이터 프래그먼트가 저장된 컨테이너의 컨테이너 식별자 정보 (container_id) 와 상기 메터데이터 프래그먼트의 식별자 정보(handle_value))를 제공한다.

<37> 서브키 인덱스(sub_key_index) 섹션(30)은 키 인덱스(key_index) 섹션(20)에서 정의된 서브키 인덱스 식별자 정보가 저장되는 'sub_index_identifier' 세그먼트, 키값들이 저장되는 'key_value' 세그먼트(14), 상기 키값을 가지는 메타데이터 프래그먼트

에 대한 식별 정보로서 메타데이터 프래그먼트가 저장된 컨테이너의 컨테이너 식별자 정보(container_id)와 프래그먼트 데이터 식별자 정보(handle_value)가 각각 저장된 'target_container' 세그먼트 및 'target_handle' 세그먼트들을 포함한다.

<38> 상기 싱글키 인덱스 정보 스트림 구조는 인덱스 정보를 예시적으로 도시한 도 7을 통해 더욱 쉽게 이해될 수 있다.

<39> 도 7은 채널 번호, 방송 시간 및 방송 재생 시간에 대한 싱글키를 포함하는 키 인덱스 리스트(key_index_list) 섹션을 도시하고 있으며, 상기 채널 번호, 방송 시간 및 방송 재생 시간에 관한 싱글키가 포함된 메타데이터 프래그먼트의 상위 노드는 도 3에 음영으로 도시된 바와 같이 'BroadcastEvent'(310)이다. 따라서, 'fragment_xpath_ptr' 세그먼트(11a) 에는 'BroadcastEvent' 프래그먼트에 대한 XPath '/TVAMain/ProgramDescription/ProgramLocationTable/BroadcastEvent'가 저장되고, 'key_xpath_ptr' 세그먼트(12a)에는 'BroadcastEvent' 프래그먼트에 대한 채널 번호, 방송 시간, 및 방송 재생 시간의 싱글키의 XPath인 '@ServiceId'(도 3의 311a), 'EventDescription/PublishedTime'(도 3의 311b) 및 'EventDescription/PublishedDuration'(도 3의 311c)이 각각 저장된다.

<40> 참고로, 도 7a는 키 인덱스 리스트(key_index_list) 섹션 중 채널 번호(싱글키의 XPath : @ServiceId)에 대한 키 인덱스(key_index) 섹션(20a) 및 서브키 인덱스(sub_key_index) 섹션(30a)을 도시하고, 도 7b는 방송 시간(싱글키의 XPath : EventDescription/PublishedTime)에 대한 키 인덱스(key_index) 섹션(20b) 및 서브키 인덱스(sub_key_index) 섹션(30b)을 도시한다.

<41> 이러한 싱글키 인덱스 정보 스트림 구조는 TV-Anytime 규격의 메타데이터 프래그먼트에 대한 특정 필드를 키로 하는 인덱스 검색 즉, 싱글키 검색만을 지원하기 때문에 하나 이상의 검색 조건 즉, 복합 조건 검색에 대해서는 효율적이지 못하다는 단점을 가진다. 예를 들면, 도 2과 같은 그리드 가이드 화면에 방송 프로그램 리스트를 표시하기 위해서는 2개 필드 즉, 채널 번호 및 방송 시간에 대한 검색 동작이 요구된다.

<42> 종래의 싱글키 인덱스 정보 스트림 구조를 이용한 복합 조건 검색을 설명하기 위해, 이하에서는 채널 범위가 507~514이고, 방송 시간이 09:30~10:00 인 프로그램 리스트를 구하는 경우를 예로 들어 설명한다. TV-Anytime 메타데이터 규격에서 이와 같은 프로그램 리스트에 관한 메타데이터를 가져오기 위한 검색 조건은 다음과 같이 표현된다.

<43> - 검색 대상 프래그먼트 (BroadcastEvent):

<44> /TVAMain/ProgramDescription/ProgramLocationTable/BroadcastEvent

<45> - 검색 조건 리스트:

<46> 507 <= ServiceId <= 514

<47> 09:30 <= EventDescription/PublishedTime <= 10:00

<48> 종래의 싱글키 인덱스 정보 스트림 구조에서, 지정된 검색 조건에 부합하는 프래그먼트를 얻기 위해서는 2가지 방법이 가능한데, 도 8를 참조하여 자세히 설명하면 다음과 같다.

<49> (1)싱글키 인덱스를 이용한 검색 방법 1

<50> 첫번째 방법은 도 8(a)에 설명된 바와 같이, ServiceId 와 EventDescription/PublishedTime에 대한 각각의 싱글키를 사용함으로써 서로 독립적으

로 각각 조건에 맞는 중간 결과의 프래그먼트 집합들을 검색한다. 다음에 그 중간 프래그먼트 집합들을 교집합(intersection)시킴으로써 조건에 맞는 최종 결과 프래그먼트 집합을 얻는 것이다.

<51> 이 방식을 도 7 및 도 8(a)를 참조하여 자세히 설명하면 다음과 같다.

<52> 먼저, 채널 번호 검색에 필요한 싱글키 정보 즉, 검색 대상 메타데이터 프래그먼트의 XPath와 싱글키의 XPath, 그리고 싱글키 값을 지정한다(S11).

<53> 메타데이터 프래그먼트의 XPath :

<54> /TVAMain/ProgramDescription/ProgramLocationTable/BroadcastEvent

<55> 채널 번호의 XPath : @ServiceId

<56> 채널 번호의 키값 : 507 <= ServiceId <= 514

<57> 다음으로, 키 인덱스 리스트(key_index_list) 섹션(10a)에서 상기 프래그먼트 XPath(11a)와 채널 번호의 XPath(12a)와 일치하는 싱글키를 찾아 채널 번호 키값의 대표값이 저장된 키 인덱스(key_index) 섹션(20a)에 대한 식별 정보를 추출하고, 추출된 식별 정보를 가진 키 인덱스(key_index) 섹션(20a)에서 검색을 원하는 싱글키의 키값(507~514)이 포함되는 키값들의 범위(500~509, 510~519)를 나타내는 대표키 값 즉, '509' 와 '519' (13a)의 대표키 값을 찾아 상기 대표키 값 '509' 와 '519' 에 관련된 키값들(500~509, 510~519)을 가지는 서브키 인덱스(sub_key_index) 섹션(14a)들에 대한 식별 정보를 추출한다. 그리고, 추출된 식별 정보를 가진 서브키 인덱스(sub_key_index) 섹션(14a)들에서 507~514의 키값들에 해당하는 메타데이터 프래그먼트 식별 정보('target_container' 세그먼트 및 'target_handle' 세그먼트에 각각 저장

된 컨테이너 식별자 정보(container_id)와 프래그먼트 데이터 식별자 정보 (handle_value))를 추출하고, 추출된 식별 정보를 이용하여 해당 메타데이터 프래그먼트를 추출한다(S12, S14).

<58> 방송 시간 검색에 필요한 싱글키 정보 즉, 검색 대상 메타데이터 프래그먼트의 XPath 정보와 싱글키의 XPath 정보, 그리고 싱글키 값은, 각각

<59> 프래그먼트의 XPath :

<60> /TVAMain/ProgramDescription/ProgramLocationTable/BroadcastEvent

<61> 방송 시간의 XPath : EventDescription/PublishedTime

<62> 방송 시간의 키값 : 09:30 <= EventDescription/PublishedTime <= 10:00

<63> 이고, 상기 채널 번호 검색의 단계와 거의 유사한 단계를 거쳐 09:30~10:00의 키값들에 해당하는 메타데이터 프래그먼트를 추출한다(S13, S15).

<64> 상기 채널번호와 방송시간에 대해 각각 추출된 메타데이터 프래그먼트들을 교집합 연산하여 공통된 메타데이터 프래그먼트의 메타데이터를 최종 결과로서 도 2와 같은 그리드 가이드 화면에 제공하게 된다(S16).

<65> (2) 싱글키 인덱스를 이용한 검색 방법 2

<66> 두번째 방법은 도 8(b)에 설명된 바와 같이 검색 조건과 관련된 2개의 싱글키중 하나(예를 들어, 채널 번호)만을 사용하여 프래그먼트들을 검색하고(S21~S23)), 해당 프래그먼트들 중에서 또다른 검색 조건인 방송 시간이 09:30에서 10:00 사이인 프래그먼트들만 골라낸다(S24).

<67> 이러한 싱글키 인덱스 정보 스트림 구조를 이용한 검색 방법들은 각각의 싱글키를 이용하여 검색함으로써 초래되는 중간 결과 프래그먼트의 갯수의 크기가 통상 매우 크기 때문에 전체 프로세스가 효율적이지 못하다. 즉, 첫번째 방법에서는 프로그램 시간대에 관계없이 해당 채널 범위의 모든 프로그램을 검색 결과로 가져오고, 모든 채널에 대하여 해당 시간대의 프로그램을 검색 결과로 가져오기 때문에 검색 결과가 매우 클 수 있다. 더욱이, 크기가 큰 두 가지의 중간 검색 결과를 병합하는 과정에서도 계산의 복잡도가 높아져서 수신장치의 오버헤드는 더욱 증가하게 된다. 두번째 방법에서는 하나의 중간 결과가 다른 검색 조건과 대비하여 추가로 필터링되어야 한다. 결과적으로, 싱글키 인덱스 정보 스트림 구조를 이용한 복합 조건 검색은 수신장치에 있어서 많은 오버헤드가 발생할 수 있는 문제의 여지가 있는 것이다.

【발명이 이루고자 하는 기술적 과제】

<68> 상술한 문제점을 해결하기 위해 제안된 본 발명은, 디지털 컨텐츠 메타데이터의 복합 조건 검색에 유용한 멀티키 인덱스 정보 스트림 구조를 제공하는데 그 목적이 있다.

<69> 또한 본 발명은, 디지털 컨텐츠 메타데이터의 복합 조건 검색에 유용한 멀티키 인덱스 정보 스트림에 대한 검색방법 및 상기 멀티키 인덱스 정보 스트림을 수신하는 수신장치를 제공하는데 그 목적이 있다.

【발명의 구성 및 작용】

<70> 상술한 목적을 달성하기 위한 본 발명에 따른 디지털 컨텐츠 메타데이터의 멀티키 인덱스 정보 스트림 구조는, 서브키 인덱스 섹션 식별 정보, 멀티키 값들 및 상기 멀티키 값들에 해당하는 메타데이터 프래그먼트의 식별 정보를 포함하는 서브키 인덱스

(sub_key_index) 섹션; 키 인덱스(key_index) 섹션 식별 정보, 상기 서브키 인덱스 (sub_key_index) 섹션들에 포함된 멀티키 값들의 범위를 나타내는 대표키 값, 및 상기 대표키 값이 나타내는 범위에 해당하는 멀티키 값을 포함하는 서브키 인덱스 섹션 식별 정보를 포함하는 키 인덱스(key_index) 섹션; 및 복합 조건 검색에 사용되는 멀티키 정보 및 키 인덱스 섹션 식별 정보를 포함하는 키 인덱스 리스트(key_index_list) 섹션을 포함한다.

<71> 또한, 상술한 목적을 달성하기 위한 본 발명에 따른 멀티키 인덱스 정보 스트림 구조를 이용한 메타데이터 검색방법은, 복합 조건 검색에 필요한 멀티키 정보 및 멀티키 값을 지정하는 단계; 상기 키 인덱스 리스트(key_index_list) 섹션에서, 상기 지정된 멀티키 정보와 일치하는 키 인덱스(key_index) 섹션들의 식별 정보를 추출하는 단계; 상기 추출된 키 인덱스(key_index) 섹션들의 식별 정보를 가지는 키 인덱스(key_index) 섹션에서, 상기 지정된 멀티키 값이 포함되는 대표키 값에 관련된 서브키 인덱스 (sub_key_index) 섹션들의 식별 정보를 추출하는 단계; 및 상기 추출된 서브키 인덱스 (sub_key_index) 섹션들의 식별 정보를 가지는 서브키 인덱스(sub_key_index) 섹션에서, 상기 지정된 멀티키 값에 해당하는 메타데이터 프래그먼트의 식별 정보를 추출하는 단계를 포함한다.

<72> 또한, 상술한 목적을 달성하기 위한 본 발명에 따른 멀티키 인덱스 정보 스트림 구조를 가지는 멀티키 인덱스 정보 스트림을 수신하는 수신장치는, 사용자의 검색 요청을 입력받고 검색 결과를 전시하는 EPG 애플리케이션; 및

<73> 상기 검색 요청에 따른 멀티키 정보 및 멀티키 값을 지정하고, 상기 지정된 멀티키 정보 및 멀티키 값에 해당하는 메타데이터 프래그먼트를 추출하여 상기 EPG 애플리케이션에 전달하는 메타데이터 엔진을 포함한다.

<74> 따라서, TVA 메타데이터를 수신하는 수신장치가 메타데이터 프래그먼트들에 대한 복합 조건 검색을 멀티키 인덱싱 기법을 이용하여 보다 효율적으로 수행할 수 있게 된다.

<75> 이와 같은 특징을 갖는 디지털 컨텐츠 메타데이터의 멀티키 인덱스 정보 스트림 구조 및 상기 멀티키 인덱스 정보 스트림 구조를 이용한 검색방법, 그리고 상기 멀티키 인덱스 정보 스트림 구조를 가지는 멀티키 인덱스 정보 스트림을 수신하는 수신장치를 첨부된 도면을 참조하여 설명하면 다음과 같다.

<76> 먼저, 전술한 데이터 컨테이너에 저장되어 오는 TVA 메타데이터 프래그먼트를 인덱스하기 위한 멀티키 인덱스 정보 스트림 구조 즉, 키 인덱스 리스트(key_index_list) 섹션(110), 키 인덱스(key_index) 섹션(120), 및 서브키 인덱스(sub_key_index) 섹션(130)의 구조를 정의하는 신텍스를 설명하고, 이후 상기 신텍스에 의해 정의되는 멀티키 인덱스 정보 스트림 구조에 대해 설명하기로 한다.

<77> 본 발명에 따른 멀티키 인덱스 정보 스트림 구조를 정의하는 신텍스는, 싱글키 인덱스 기술문헌에 정의된 신텍스와 달리 key_descriptor(), high_key_value_descriptor(), key_value_descriptor()와 같은 멀티키 인덱싱 개념을 위해 새롭게 도입된 구조를 포함하여, 키 인덱스 리스트(key_index_list) 섹션, 키 인덱스(key_index) 섹션, 및 서브키 인덱스(sub_key_index) 섹션의 구조를 재편성한다.

<78> 1. 키 인덱스 리스트(key_index_list) 섹션

<79> 키 인덱스 리스트(key_index_list) 섹션은 전송되는 모든 멀티키의 리스트를 제공한다. 각각의 키 인덱스 리스트(key_index_list) 구조 안에는 표 1에 제시된 바와 같이 멀티키 인덱싱이 가능하도록 key_descriptor()를 포함한다.

<80> 【표 1】

신택스	비트수(변경가능)
key_index_list() {	
for (j=0; j<key_index_count; j++) {	
fragment_xpath_ptr	16
key_descriptor()	
index_container	16
key_index_identifier	8
}	
}	

<81> key_index_count: 전송되는 모든 멀티키의 수 즉, 전체 XML 문서에 대한 인덱스 수를 상술한다.

<82> fragment_xpath_ptr(): 메타데이터 프래그먼트를 나타내는 프래그먼트의 XPath.

<83> key_descriptor(): 인덱싱될 목표 프래그먼트 그룹의 XPath 위치내에서 관련 XPath 위치와 같은, 인덱싱을 위한 멀티키의 정보 및 멀티키를 구성하는 각 엘리먼트/속성에 대한 인코딩 표시자의 정보를 기술한다.

<84> index_container: 지정된 키 인덱스(key_index) 섹션이 존재하는 컨테이너를 식별한다.

<85> key_index_identifier: index_container에 의해 상술된 컨테이너내에서 키 인덱스(key_index) 섹션을 식별한다. index_container와 key_index_identifier의 조합에 의해 지정된 키 인덱스(key_index) 섹션이 유일하게 식별될 수 있다 .

<86> 2 . 키 디스크립터(key_descriptor)

<87> 멀티키는 복합된 키 속성이다. 멀티키를 구성하는 각각의 키 속성에 대하여, key_descriptor는 키의 xpath 위치 등의 특성을 기술한다. 아래의 표 2는 key_descriptor를 나타낸다.

<88> 【표 2】

신택스	비트수(변경가능)
key_descriptor () {	
key_attribute_count	8
for (j=0; j<key_attribute_count; j++) {	
key_xpath_ptr	16
}	
}	

<89> key_attribute_count: 멀티키를 구성하는 키 속성의 수를 상술한다.

<90> key_xpath_ptr: 멀티키로서 사용되는 노드의 fragment_xpath_ptr에 관련한 상대 경로이다.

<91> 3 . 키 인덱스(key_index) 섹션

<92> high_key_value_descriptor()가 새롭게 도입된다.

<93> high_key_value_descriptor()는 키 인덱스(key_index) 섹션이 가리키는 서브키 인덱스(sub_key_index) 섹션의 갯수(sub_index_count) 만큼의 서브키 인덱스(sub_key_index) 섹션들에 대해 해당 서브키 인덱스(sub_key_index) 섹션내의 멀티키 값들중 가장 큰 멀티키 값을 지정한다.

<94>

【표 3】

신택스	비트수(변경가능)
key_index () {	
key_index_identifier	8
sub_index_count	8
for (j=0; j< sub_index_count; j++) {	
high_key_value_descriptor()	16 * key_attribute_count
sub_index_container	16
sub_index_identifier	8
}	
}	

<95> key_index_identifier: index_container에 의해 상술된 컨테이너내에서 키 인덱스(key_index) 섹션을 식별한다. index_container와 key_index_identifier의 조합에 의해 키 인덱스(key_index) 섹션이 유일하게 식별될 수 있다. 이것은 키 인덱스 리스트(key_index_list) 섹션에서 정의된다.

<96> sub_index_container: 지정된 서브키 인덱스(sub_key_index) 섹션이 존재하는 컨테이너를 식별한다.

<97> sub_index_identifier: sub_index_container에 의해 상술된 컨테이너내에서 서브키 인덱스(sub_key_index) 섹션을 식별한다. sub_index_container와 sub_index_identifier의 조합에 의해 서브키 인덱스(sub_key_index) 섹션이 유일하게 식별될 수 있다.

<98> 표 4는 high_key_value_descriptor()를 나타낸다.

<99>

【표 4】

신택스	비트수(변경가능)
high_key_value_descriptor() {	
for (j=0; j< key_attribute_count; j++) {	
key_attribute_value	16
}	
}	

<100> key_attribute_count: 멀티키를 구성하는 키 속성의 수를 상술한다. 이것은 키 인덱스 리스트(key_index_list) 섹션에서 정의된다.

<101> key_attribute_value: 각각의 키 속성에 대한 대표키 값을 나타낸다. 상기 값 인코딩 형태는 싱글키 인덱스 기술문헌의 key_value와 같다.

<102> high_key_value_descriptor()가 멀티키 값을 가질 경우 멀티키 값 사이의 크기 비교는 다음과 같다. 멀티키 값은 사전 편찬 방식에 의해 순서가 결정된다. 즉, k1, k2, ..., kn 의 키들로 구성된 멀티키 (k1, k2, ..., kn)에 대해, k1이 가장 큰 우선순위를 가지고 kn은 가장 작은 우선순위를 가진다고 가정할 때, 두 멀티키 값(a1, a2, ..., an) 와 멀티키 값(b1, b2, ..., bn)에 대하여,

<103> * 모든 j (0≤j≤i-1)에 대해 aj=bj 및 ai>bi가 되도록 정수 i (0≤i≤n-1)가 존재하는 경우에 한해서 멀티키 값(a1, a2, ..., an)이 멀티키 값(b1, b2, ..., bn)보다 더 크다.

<104> * 모든 j (0≤j≤i-1)에 대해 aj=bj 및 ai<bi가 되도록 정수 i (0≤i≤n-1)가 존재하는 경우에 한해서 멀티키 값(a1, a2, ..., an)이 멀티키 값(b1, b2, ..., bn)보다 더 작다.

<105> * 모든 i (1≤i≤n)에 대해 ai=bi인 경우에 한해서 멀티키 값(a1, a2, ..., an)은 멀티키 값(b1, b2, ..., bn)과 같다.

<106> 4 . 서브 키 인덱스(sub_key_index) 섹션

<107> key_value_descriptor()가 멀티키 인덱싱 개념을 위해 새롭게 도입된다. key_value_descriptor()는 지시하는 해당 프래그먼트의 멀티키 값을 나타낸다.

<108> 【표 5】

신택스	비트수(변경가능)
sub_key_index () {	
sub_index_identifier	8
reference_count	8
for (j=0; j< reference_count; j++) {	
key_value_descriptor()	16 * key_attribute_count
target_container	16
target _handle	16
}	
}	

<109> sub_index_identifier: sub_index_container에 의해 식별된 컨테이너내에서 서브 키 인덱스(sub_key_index) 섹션을 식별한다. sub_index_container와 sub_index_identifier의 조합에 의해 서브키 인덱스(sub_key_index) 섹션이 유일하게 식별될 수 있다. 이것은 키 인덱스(key_index) 섹션에서 정의된다.

<110> reference_count: sub_key_index()내에 포함된 멀티키 값들의 갯수를 상술한다.

<111> target_container: 지정된 메타데이터 프래그먼트가 존재하는 컨테이너를 식별한다.

<112> target_handle: target_container 에 의해 식별된 컨테이너내에서 메타데이터 프래그먼트 섹션을 식별한다. target_container와 target_handle의 조합에 의해 메타데이터 프래그먼트 섹션이 유일하게 식별될 수 있다 .

<113> 표 6은 key_value_descriptor()를 나타낸다.

<114> 【표 6】

신택스	비트수(변경가능)
key_value_descriptor() {	
for (j=0; j< key_attribute_count; j++) {	
key_attribute_value	16
}	
}	

<115> key_attribute_count: 멀티키를 구성하는 속성의 수를 상술하며, 키 인덱스 리스트(key_index_list) 섹션에서 정의된다.

<116> key_attribute_value: 각각의 키 속성을 표현한다. 그 형태는 싱글키 인덱스 기술 문헌의 key_value와 동일하다.

<117> key_value_descriptor() 값 사이의 비교는 키 인덱스(key_index) 섹션 구조에서의 high_key_value_descriptor() 값 사이의 비교와 동일하다.

<118> 이하, 전술한 신텍스에 의해 정의되는 멀티키 인덱스 정보 스트림 구조를 인덱스 정보 스트림상의 세그먼트로 표현한 도 9을 참조하여 상기 구조를 살펴보기로 한다.

<119> 멀티키 인덱스 정보 스트림 구조에서 정의되는 키 인덱스 리스트(key_index_list) 섹션(110)은, 전송되는 모든 멀티키의 리스트를 제공한다. 이 리스트에는 각 멀티키들에 대한 멀티키 정보(즉, 멀티키가 포함된 메타데이터 프래그먼트에 대한 XPath(fragment_xpath_ptr)와 멀티키로서 사용되는 노드들의 상기 메타데이터 프래그먼

트의 XPath 위치에서의 관련 경로 즉, 멀티키로 사용되는 노드들의 XPath(key_descriptor))와, 후술되는 키 인덱스(key_index) 섹션(120)에 대한 식별 정보가 포함된다. 싱글키 인덱스 정보 스트림 구조와 마찬가지로, 상기 메타데이터 프래그먼트의 XPath는 TVA 메타데이터 XML 문서의 루트 노드에 대한 경로 즉, 절대 경로이며, 멀티키로 사용되는 노드의 XPath 즉, 멀티키의 XPath는 상기 메타데이터 프래그먼트에 대한 멀티키의 상대 경로를 나타낸다. 메타데이터 프래그먼트에 대한 XPath와 멀티키에 대한 XPath는 각각 'fragment_xpath_ptr' 세그먼트(111)와 'key_descriptor' 세그먼트(112)에 저장된다.

<120> 또한, 키 인덱스 리스트(key_index_list) 섹션(110)에는 후술되는 각 멀티키의 키 인덱스(key_index) 섹션(120)에 대한 식별 정보(즉, 키 인덱스(key_index) 섹션(120)이 저장된 컨테이너의 컨테이너 식별자 정보(container_id) 및 키 인덱스 식별자 정보)가 포함되는데, 상기 컨테이너 식별자 정보 및 키 인덱스 식별자 정보들은 각각 키 인덱스 리스트(key_index_list) 섹션(110)의 'index_container' 세그먼트 및 'key_index_identifier' 세그먼트에 저장되어 전송된다.

<121> 멀티키 인덱스 정보 스트림 구조에서 정의되는 키 인덱스(key_index) 섹션(120)은, 후술되는 모든 서브키 인덱스(sub_key_index) 섹션 (130)의 리스트를 제공하는데, 이 리스트에는 각 서브키 인덱스(sub_key_index) 섹션(130)들에 포함된 멀티키 값들의 범위를 나타내는 정보 즉, 각 서브키 인덱스(sub_key_index) 섹션(130)내 멀티키 값 중 가장 큰 멀티키 값(이하 '대표키 값')과, 각 대표키 값에 관련된 서브키 인덱스 섹션(130)에 대한 식별 정보(즉, 서브키 인덱스(sub_key_index) 섹션이 저장된 컨테이너의 컨테이너 식별자 정보(container_id) 및 서브키 인덱스 식별자 정보)가 포함된다. 본 실시예에

서 멀티키 값 사이의 크기 비교 방법은 표 4에 대한 설명에서 언급한 멀티키 값들의 크기 비교 방법과 동일하다.

<122> 키 인덱스(key_index) 섹션(120)은 키 인덱스 리스트(key_index_list) 섹션(110)에서 정의된 키 인덱스 식별자 정보가 저장되는 'key_index_identifier' 세그먼트, 각 서브키 인덱스(sub_key_index) 섹션(130)의 대표키 값이 저장된 'high_key_value_descriptor' 세그먼트(113), 대표키 값이 나타내는 범위에 해당하는 멀티키 값들을 포함하는 서브키 인덱스(sub_key_index) 섹션(130)에 대한 식별 정보로서 서브키 인덱스(sub_key_index) 섹션(130)이 저장된 컨테이너의 컨테이너 식별자 정보(container_id)와 서브키 인덱스 식별자 정보가 각각 저장된 'sub_index_container' 세그먼트 및 'sub_index_identifier' 세그먼트들을 포함한다.

<123> 멀티키 인덱스 정보 스트림 구조에서 정의되는 서브키 인덱스(sub_key_index) 섹션(130)은, 해당 서브키 인덱스(sub_key_index) 섹션(130)에 포함된 멀티키 값들에 대한 리스트를 제공하는데, 이 리스트에는 해당 서브키 인덱스 섹션(130)에 포함된 멀티키 값들 및 상기 멀티키 값들을 가지는 메타데이터 프래그먼트에 대한 식별 정보(즉, 메타데이터 프래그먼트가 저장된 컨테이너의 컨테이너 식별자 정보 (container_id) 와 상기 메터데이터 프래그먼트의 식별자 정보(handle_value))를 제공한다.

<124> 서브키 인덱스(sub_key_index) 섹션(130)은 키 인덱스(key_index) 섹션(120)에서 정의된 서브키 인덱스 식별자 정보가 저장되는 'sub_index_identifier' 세그먼트, 멀티키 값들이 저장되는 'key_value_decriptor' 세그먼트(114), 멀티키 값을 가지는 메타데이터 프래그먼트에 대한 식별 정보로서 메타데이터 프래그먼트가 저장된 컨테이너의 컨테이너 식별자 정보(container_id)와 프래그먼트 데이터 식별자 정보(handle_value)

가 각각 저장된 'target_container' 세그먼트 및 'target_handle' 세그먼트들을 포함한다.

<125> 상기 멀티키 인덱스 정보 스트림 구조는 인덱스 정보를 예시적으로 도시한 도 10을 통해 더욱 쉽게 이해될 수 있다.

<126> 도 10은 채널 번호 및 방송 시간에 대한 멀티키를 포함하는 멀티키 인덱스 리스트(key_index_list) 섹션을 도시하고 있으며, 상기 도면에 도시된 상기 채널 번호 및 방송 시간에 관한 멀티키가 포함된 메타데이터 프래그먼트의 상위 노드는 도 3에 음영으로 도시된 바와 같이 'BroadcastEvent'(310)이다. 따라서, 'fragment_xpath_ptr' 세그먼트(111) 에는 'BroadcastEvent' 프래그먼트에 대한 XPath '/TVAMain/ProgramDescription/ProgramLocationTable/BroadcastEvent'가 저장되고, 'key_descriptor' 세그먼트(112)에는 'BroadcastEvent' 프래그먼트에 대한 채널 번호 및 방송 시간의 멀티키의 XPath인 '@ServiceId'(311a) 및 'EventDescription/PublishedTime'(311b)가 각각 저장된다.

<127> 이러한 멀티키 인덱스 정보 스트림 구조는 하나 이상의 검색 조건 즉, 복합 조건 검색시, 메타데이터 프래그먼트에 대한 검색과 접근을 효율적으로 처리할 수 있다.

<128> 본 실시예에서는 채널 번호 및 방송 시간에 대한 멀티키를 예시하였으나, 여러가지 다른 멀티키들이 조합되어 사용될 수 있다. 예를 들면, 방송 스케쥴에 관련하여 프로그램의 시작 시간과 종료 시간에 대한 멀티키, 프로그램내 인물(배우, 감독 등)의 성(姓)과 이름에 대한 멀티키 등이 사용될 수 있다.

<129> 방송 스케쥴에 관련하여 프로그램의 시작 시간과 종료 시간에 대한 멀티키가 사용될 경우, 방송 스케쥴에 관련하여 프로그램의 시작 시간과 종료 시간에 대한 멀티키가 포함된 메타데이터 프래그먼트의 상위 노드는 미도시된 'Schedule' 이다. 따라서, 'fragment_xpath_ptr' 세그먼트(111) 에는 'Schedule' 프래그먼트에 대한 XPath '/TVAMain/ProgramDescription/ProgramLocationTable/Schedule' 이 저장되고, 'key_descriptor' 세그먼트(112)에는 'Schedule' 프래그먼트에 대한 프로그램의 시작 시간과 종료 시간의 멀티키의 XPath인 '@start' 및 '@end' 가 각각 저장된다.

<130> 또한, 프로그램내 인물(배우, 감독 등)의 성(姓)과 이름에 대한 멀티키가 사용될 경우, 프로그램내 인물의 성(姓)과 이름에 대한 멀티키가 포함된 메타데이터 프래그먼트의 상위 노드는 미도시된 'PersonName' 이다. 따라서, 'fragment_xpath_ptr' 세그먼트(111) 에는 'PersonName' 프래그먼트에 대한 XPath '/TVAMain/ProgramDescription/CreditsInformationTable/PersonName' 이 저장되고, 'key_descriptor' 세그먼트(112)에는 'PersonName' 프래그먼트에 대한 프로그램내 인물의 성(姓)과 이름에 대한 멀티키의 XPath인 'FamilyName' 및 'GivenName' 이 각각 저장된다.

<131> 이하에서는 도 11을 참조하여, 멀티키 인덱스 정보 스트림 구조에서 하나 이상의 검색 조건에 부합하는 프래그먼트를 얻기 위한 검색 방법을 설명한다.

<132> 먼저 하나 이상의 검색 조건을 수행하기 위한 멀티키 정보 및 멀티키 값을 지정한다(S100). 여기서 멀티키 정보는 전술한 바와 같이 멀티키가 인덱싱할 메타데이터 프래그먼트의 절대 경로 정보 즉, 상기 메타데이터 프래그먼트의 XPath(fragment_xpath_ptr)와 상기 메타데이터 프래그먼트에 대한 멀티키의 상대 경로 정보(상기 프래그먼트의

XPath 위치에서의 관련 경로) 즉, 멀티키로서 사용되는 노드들의 XPath(key_descriptor)를 말한다.

<133> 지정된 멀티키 정보 및 멀티키 값에 해당하는 메타데이터 프래그먼트를 찾기 위해, 전송 스트림의 키 인덱스 리스트(key_index_list) 섹션(110)에서 지정된 멀티키 정보와 일치하는 키 인덱스(key_index) 섹션(120)의 식별 정보를 추출한다(S200).

<134> 다음 단계로서, 추출된 키 인덱스 섹션 식별 정보를 이용하여 키 인덱스(key_index) 섹션(120)을 찾아, 상기 지정된 멀티키 값이 포함되는 대표키 값에 관련된 서브키 인덱스(sub_key_index) 섹션(130)의 식별 정보를 추출한다(S300).

<135> 다음 단계로서, 추출된 서브키 인덱스(sub_key_index) 섹션의 식별 정보를 이용하여 서브키 인덱스(sub_key_index) 섹션(130)을 찾은 후, 상기 지정된 멀티키 값에 해당하는 메타데이터 프래그먼트의 식별 정보를 추출한다(S400).

<136> 마지막으로, 추출된 메타데이터 프래그먼트의 식별 정보를 이용하여 해당 데이터 컨테이너에서 메타데이터 프래그먼트를 추출하고(S500), 추출된 메타데이터프래그먼트에 따라 검색 결과를 출력한다.

<137> 상기 도 11에 따른 검색방법을 도 7 및 도 8를 참조하여 설명한 채널 번호 및 방송 시간에 대한 검색에 적용하면 다음과 같다.

<138> 메타데이터를 가져오기 위한 검색 조건은 아래와 같다.

<139> - 검색 대상 프래그먼트 (BroadcastEvent):

<140> /TVAMain/ProgramDescription/ProgramLocationTable/BroadcastEvent

<141> - 검색 조건 리스트:

<142> 507 <= ServiceId <= 514

<143> 9:30 <= EventDescription/PublishedTime <= 10:00

<144> 이하 도 10을 참조하여 자세히 설명한다.

<145> 먼저, 채널 번호 및 방송 시간 검색에 필요한 멀티키 정보 즉, 검색 대상 메타데이터 프래그먼트의 XPath와 상기 메타데이터 프래그먼트에 대한 멀티키의 XPath, 그리고 멀티키 값을 지정한다(S11).

<146> 프래그먼트의 XPath :

<147> /TVAMain/ProgramDescription/ProgramLocationTable/BroadcastEvent

<148> 채널 번호의 XPath : @ServiceId

<149> 방송 시간의 XPath : EventDescription/PublishedTime

<150> 채널 번호의 키 값 : 507 <= ServiceId <= 514

<151> 방송 시간의 키 값 : 9:30 <= EventDescription/PublishedTime <= 10:00

<152> 다음으로, 키 인덱스 리스트(key_index_list) 섹션(110)에서 상기 프래그먼트 XPath(111)와 채널 번호/방송 시간의 XPath(112)와 일치하는 멀티키를 찾아 채널 번호/방송 시간 키 값의 대표값이 저장된 키 인덱스(key_index) 섹션(120)에 대한 식별 정보를 추출한다. 본 실시예에서 채널 번호는 방송 시간보다 큰 우선 순위를 가진다. 추출된 식별 정보를 가진 키 인덱스(key_index) 섹션(120)에서 검색을 원하는 멀티키 값(507~514/09:30~10:00)이 포함되는 멀티키 값들의 범위(500~509/09:10~10:00(114a), 510~519/09:10~10:00(114b))를 나타내는 대표키 값 즉, '509/10:00' (113a) 와 '519/10:00' (113b) 의 대표키 값을 찾아 대표키 값이 나타내는 멀티키 값들을 가지는 서

브키 인덱스(sub_key_index) 섹션(114a, 114b)들에 대한 식별 정보를 추출하고, 추출된 식별 정보를 가진 서브키 인덱스(sub_key_index) 섹션(114a, 114b)들에서 원하는 멀티키 값(507~514/09:30~10:00)에 해당하는 멀티키 값들 즉, '507/09:30', '507/09:40', …, '509:10:00' 의 키 값들과 '510/09:30', '510/09:40' ,…, '514:10:00 ' 의 키 값들에 해당하는 메타데이터 프래그먼트 식별 정보('target_container' 세그먼트 및 'target_handle' 세그먼트에 각각 저장된 컨테이너 식별자 정보(container_id)와 프래그먼트 데이터 식별자 정보(handle_value))를 추출하고, 추출된 식별 정보를 이용하여 해당 메타데이터 프래그먼트를 추출한다.

<153> 도 12는 본 발명에 따른 멀티키 인덱스 정보 스트림을 수신하는 수신장치를 도시한다.

<154> 상기 수신장치는, 사용자의 검색 요청을 입력받아 메타데이터 엔진(500)에 전달하고, 메타데이터 엔진(500)으로부터 검색 결과를 수신하여 전시하는 EPG 애플리케이션(400), 메타데이터 검색에 필요한 멀티키 정보 및 멀티키 값을 지정하고, 지정된 멀티키 정보 및 멀티키 값에 해당하는 메타데이터 프래그먼트를 전송 스트림(600)으로부터 추출하고, 추출된 메타데이터 프래그먼트를 처리하여 EPG 애플리케이션(400)에 출력하는 메타데이터 엔진(500)으로 구성된다.

<155> 상기 메타데이터 엔진(500)은,

<156> 입력된 멀티키 정보 및 멀티키 값에 해당하는 메타데이터 프래그먼트의 식별 정보를 전송 스트림(600)으로부터 추출하는 인덱스 매니저(510),

<157> 입력된 메타데이터 프래그먼트의 식별 정보에 해당하는 메타데이터 프래그먼트를 전송 스트림(600)으로부터 추출하는 프래그먼트 매니저(520), 및

<158> 메타데이터 검색에 필요한 멀티키 정보 및 멀티키 값을 지정하고, 지정된 멀티키 정보 및 멀티키 값을 인덱스 매니저(510)로 전달하고, 인덱스 매니저(510)에서 추출된 메타데이터 프래그먼트의 식별 정보를 프래그먼트 매니저(520)로 전달하고, 그리고, 프래그먼트 매니저(520)에서 추출된 메타데이터 프래그먼트를 처리하여 검색 결과를 출력하는 질의 처리부(530)를 포함한다.

<159> 도 13은 도 12의 수신장치에 의한 멀티키 인덱스 정보 스트림에 대한 검색방법을 도시한다.

<160> EPG 애플리케이션(400)을 통해 사용자가 하나 이상의 복합 조건에 대한 메타데이터 검색을 요청하면(S1100), 질의처리부가(530)가 검색에 필요한 멀티키 정보 및 멀티키 값을 지정한다(S1200).

<161> 질의처리부(530)에서 지정된 멀티키 정보 및 멀티키 값은 인덱스 매니저(510)로 입력되고, 인덱스 매니저(510)는 지정된 멀티키 정보 및 멀티키 값을 이용하여 전송 스트림(600)에서 키 인덱스 리스트(key_index_list) 섹션, 키 인덱스(key_index) 섹션 및 서브키 인덱스(sub_key_index) 섹션을 수신하고, 지정된 멀티키 값에 해당하는 메타데이터 프래그먼트의 식별 정보를 추출한다(S1300).

<162> 추출된 메타데이터 프래그먼트의 식별 정보는 질의처리부(530)을 통해, 또는 질의처리부(530)를 거치지 않고 직접 프래그먼트 매니저(520)로 전달된다(S1400).

<163> 다음 단계로서, 프래그먼트 매니저(520)는 상기 추출된 메타데이터 프래그먼트의 식별 정보를 이용하여 전송 스트림(600)에서 메타데이터 프래그먼트를 추출하여 질의처리부(530)로 출력한다.

<164> 마지막으로, 질의처리부(530)는 메타데이터 프래그먼트를 처리하여(S1600) 검색결과를 EPG 애플리케이션(400)으로 출력하고, EPG 애플리케이션(400)은 사용자에게 전시한다(S1700)

<165> 본 발명은 도면에 도시된 일 실시예를 참고로 설명되었으나 이는 예시적인 것에 불과하며, 본 기술분야의 통상의 지식을 가진 자라면 이로부터 다양한 변형 및 균등한 타 실시예가 가능하다는 점을 이해할 것이다. 따라서, 본 발명의 진정한 기술적 보호범위는 첨부된 청구범위의 기술적 사상에 의해 정해져야 할 것이다.

【발명의 효과】

<166> 본 발명은 TV-Anytime 메타데이터가 프래그먼트 단위로 전송되는 전송 스트림 환경에서, 메타데이터 프래그먼트의 보다 효율적인 검색 및 접근 기능을 제공하기 위한 멀티키 인덱스 정보 스트림 구조, 멀티키 인덱스 정보 스트림에 대한 검색 방법 및 멀티키 인덱스 정보 스트림을 수신하는 수신장치를 제시하였다.

<167> 본 발명을 통하여, TV-Anytime 메타데이터에 대한 복합 조건에 대한 동시 검색이 가능하고, TV-Anytime 메타데이터에 대한 복합 조건 검색이 수행될 경우, 수신장치의 오버헤드가 감소하여 검색 시간이 단축되고 수신장치의 효율이 증대된다.

【특허청구범위】

【청구항 1】

서브키 인덱스 섹션 식별 정보, 멀티키 값들, 및 상기 멀티키 값들에 해당하는 메타데이터 프래그먼트의 식별 정보를 포함하는 서브키 인덱스(sub_key_index) 섹션;

키 인덱스 섹션 식별 정보, 상기 서브키 인덱스(sub_key_index) 섹션들에 포함된 멀티키 값들의 범위를 나타내는 대표키 값, 및 상기 대표키 값이 나타내는 범위에 해당하는 멀티키 값을 포함하는 서브키 인덱스 섹션 식별 정보를 포함하는 키 인덱스 (key_index) 섹션; 및

복합 조건 검색에 사용되는 멀티키 정보 및 키 인덱스 섹션 식별 정보를 포함하는 키 인덱스 리스트(key_index_list) 섹션을 포함하는 것을 특징으로 하는 디지털 컨텐츠 메타데이터의 복합 조건 검색을 위한 멀티키 인덱스 정보 스트림 구조.

【청구항 2】

제 1항에 있어서, 상기 멀티키 정보는, 메타데이터 프래그먼트의 XPath 및 멀티키의 XPath인 것을 특징으로 하는 디지털 컨텐츠 메타데이터의 복합 조건 검색을 위한 멀티키 인덱스 정보 스트림 구조.

【청구항 3】

제 1항에 있어서, 상기 멀티키가 k1이 가장 큰 우선순위를 가지고 kn이 가장 작은 우선순위를 가지는 k1, k2, ..., kn 의 키들로 구성되는 멀티키(k1, k2, ..., kn)일 경우, 상기 멀티키 값의 크기 비교는 두 멀티키 값 (a1, a2, ..., an) 와 (b1, b2, ..., bn)에 대하여 , 모든 j (0≤j≤i-1) 에 대해 aj=bj 및 ai>bi가 되도록 정수 i (0≤i≤

n-1)가 존재하는 경우에 한해서 멀티키 값(a1, a2, ..., an)이 멀티키 값(b1, b2, ..., bn)보다 더 크고, 모든 j (0≤j≤i-1)에 대해 aj=bj 및 ai<bi가 되도록 정수 i (0≤i≤n-1)가 존재하는 경우에 한해서 멀티키 값(a1, a2, ..., an)이 멀티키 값(b1, b2, ..., bn)보다 더 작고, 모든 i (1≤i≤n)에 대해 ai=bi인 경우에 한해서 멀티키 값(a1, a2, ..., an)은 멀티키 값(b1, b2, ..., bn)과 같은 것을 특징으로 하는 디지털 컨텐츠 메타데이터의 복합 조건 검색을 위한 멀티키 인덱스 정보 스트림 구조.

【청구항 4】

제 1항 내지 제 3항중 어느 한 항에 따른 멀티키 인덱스 정보 스트림 구조를 이용한 메타데이터 검색방법에 있어서,

복합 조건 검색에 필요한 멀티키 정보 및 멀티키 값을 지정하는 단계;

상기 키 인덱스 리스트(key_index_list) 섹션에서, 상기 지정된 멀티키 정보와 일치하는 키 인덱스(key_index) 섹션들의 식별 정보를 추출하는 단계;

상기 추출된 키 인덱스(key_index) 섹션들의 식별 정보를 가지는 키 인덱스(key_index) 섹션에서, 상기 지정된 멀티키 값이 포함되는 대표키 값에 관련된 서브키 인덱스(sub_key_index) 섹션들의 식별 정보를 추출하는 단계; 및

상기 추출된 서브키 인덱스(sub_key_index) 섹션들의 식별 정보를 가지는 서브키 인덱스(sub_key_index) 섹션에서, 상기 지정된 멀티키 값에 해당하는 메타데이터 프래그먼트의 식별 정보를 추출하는 단계를 포함하는 것을 특징으로 하는 멀티키 인덱스 정보 스트림 구조를 이용한 메타데이터 검색방법

【청구항 5】

제 1항 내지 제 3항중 어느 한 항에 따른 멀티키 인덱스 정보 스트림 구조를 가지는 멀티키 인덱스 정보 스트림을 수신하는 수신장치에 있어서,

사용자의 검색 요청을 입력받고 검색 결과를 전시하는 EPG 애플리케이션; 및

상기 검색 요청에 따른 멀티키 정보 및 멀티키 값을 지정하고, 상기 지정된 멀티키 정보 및 멀티키 값에 해당하는 메타데이터 프래그먼트를 추출하여 상기 EPG 애플리케이션에 전달하는 메타데이터 엔진을 포함하는 것을 특징으로 하는 멀티키 인덱스 정보 스트림 수신장치.

【청구항 6】

제 5항에 있어서, 상기 메타데이터 엔진은,

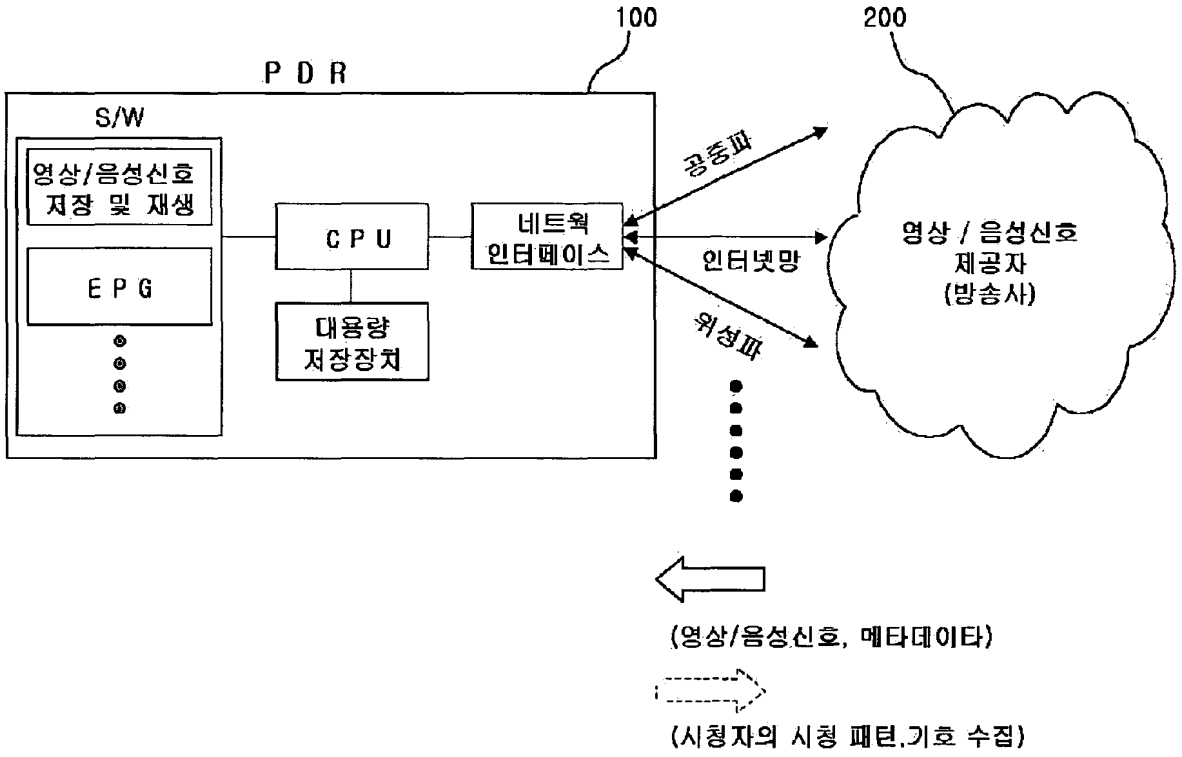
입력된 멀티키 정보 및 멀티키 값에 해당하는 메타데이터 프래그먼트의 식별 정보를 추출하는 인덱스 매니저;

입력된 메타데이터 프래그먼트의 식별 정보에 해당하는 메타데이터 프래그먼트를 추출하는 프래그먼트 매니저; 및,

상기 멀티키 정보 및 멀티키 값을 지정하고, 상기 지정된 멀티키 정보 및 멀티키 값을 상기 인덱스 매니저로 전달하고, 상기 인덱스 매니저에서 추출된 메타데이터 프래그먼트의 식별 정보를 상기 프래그먼트 매니저로 전달하고, 그리고, 상기 프래그먼트 매니저에서 추출된 메타데이터 프래그먼트를 처리하여 상기 EPG 애플리케이션으로 검색 결과를 출력하는 질의 처리부를 포함하는 것을 특징으로 하는 멀티키 인덱스 정보 스트림 수신장치.

【도면】

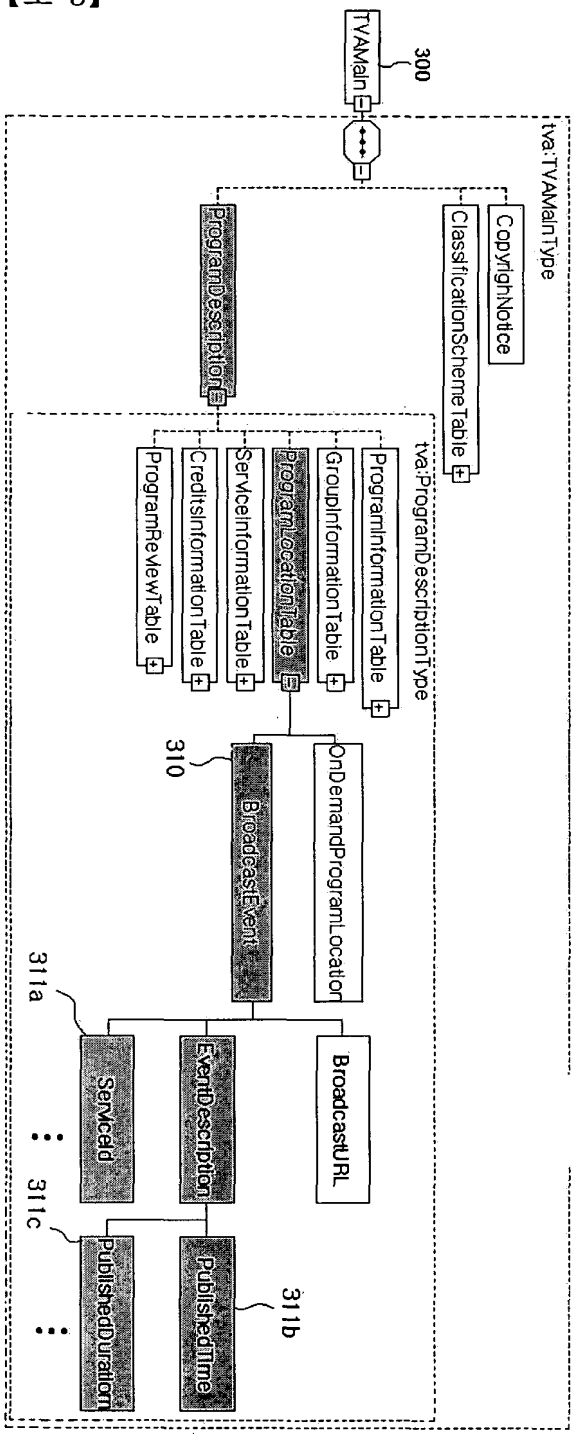
【도 1】


100
200
PDR
S/W
영상/음성신호 저장 및 재생
EPG
CPU
네트웍 인터페이스
대용량 저장장치
공중파
인터넷망
위성파
영상 / 음성신호 제공자 (방송사)
(영상/음성신호, 메타데이타)
(시청자의 시청 패턴,기호 수집)

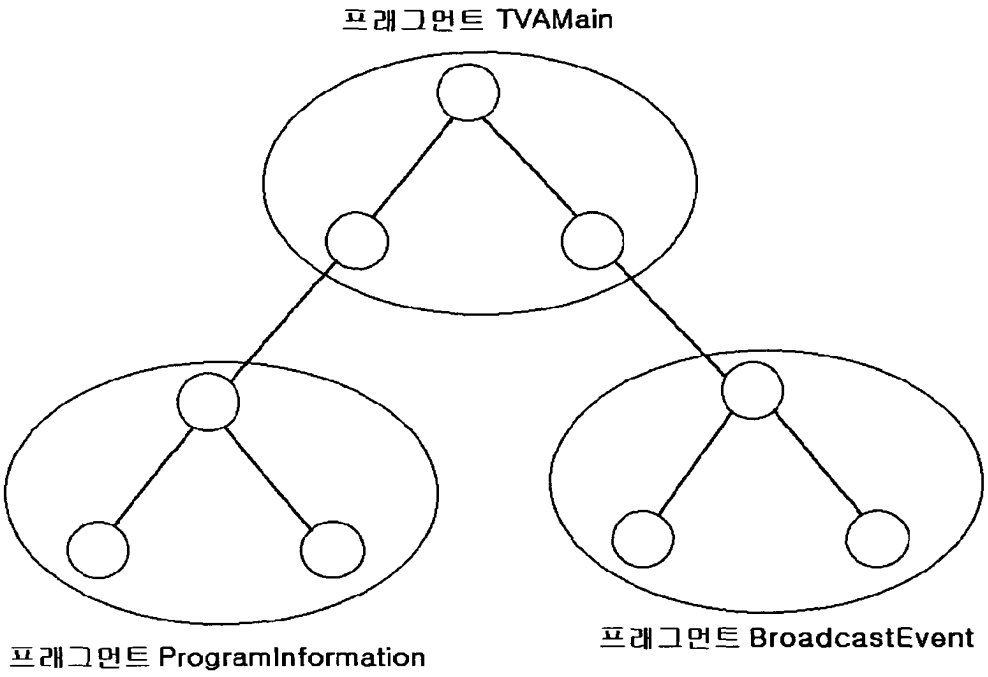
【도 2】


Today
9:00p
9:30p
10:00p
507 HBOF
Out of Africa - Movie - Spanish HD
508 HBFW
Into the West - Movie
Hocus Pocus
512 MAX
Vertical Limit - Movie HD
American Be...
513 MMAX
The Gauntlet - Movie
514 MaxW
The Bridges of Madison County - Movie LB

【도 3】

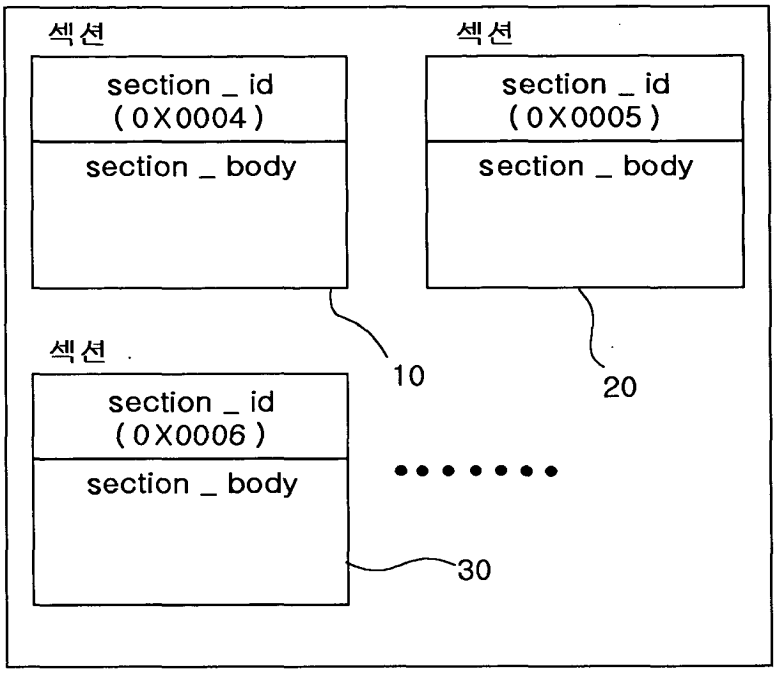

TVAMain
300
tva:TVAMainType
CopyrightNotice
ClassificationSchemeTable
ProgramDescription
tva:ProgramDescriptionType
ProgramInformationTable
GroupInformationTable
ProgramLocationTable
ServiceInformationTable
CreditsInformationTable
ProgramReviewTable
310
BroadcastEvent
OnDemandProgramLocation
311a
ServiceId
EventDescription
BroadcastURL
311c
PublishedDuration
PublishedTime
311b

【도 4】

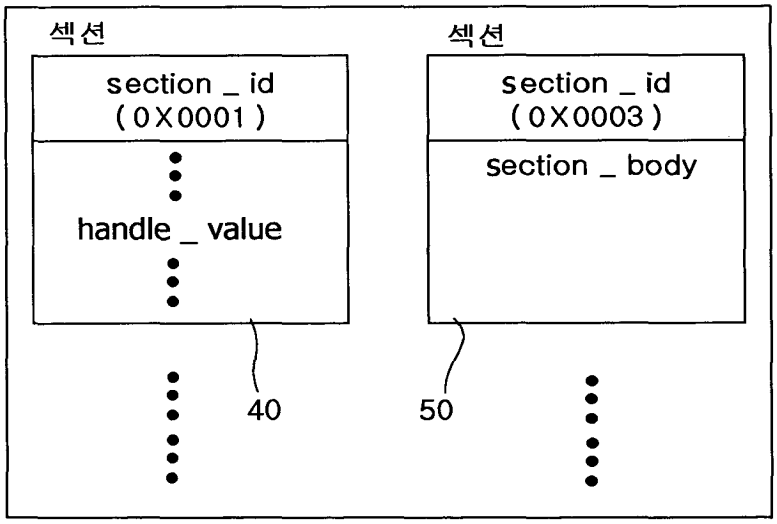

프래그먼트 TVAMain
프래그먼트 ProgramInformation
프래그먼트 BroadcastEvent

【도 5】

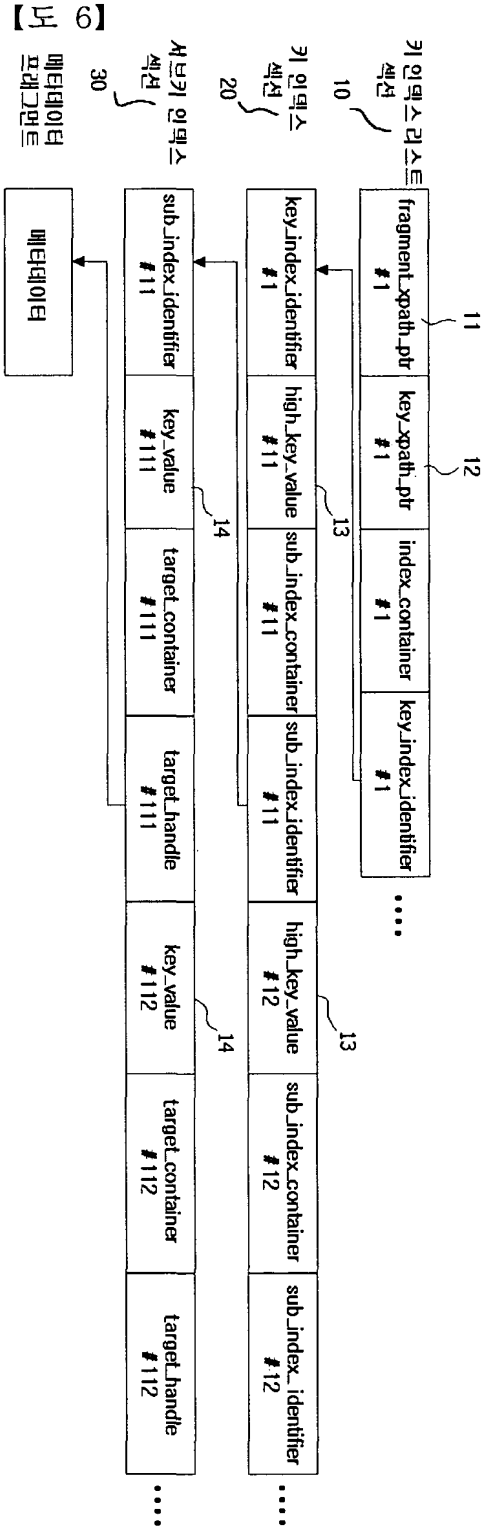
인덱스 컨테이너



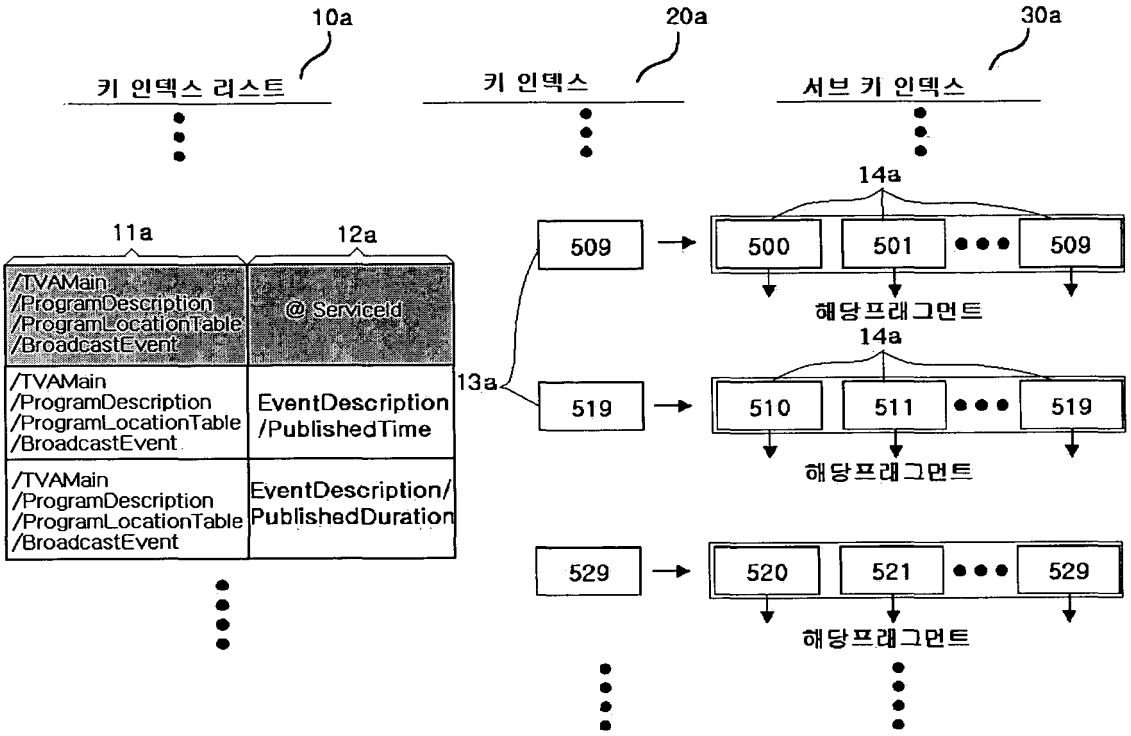
데이터 컨테이너



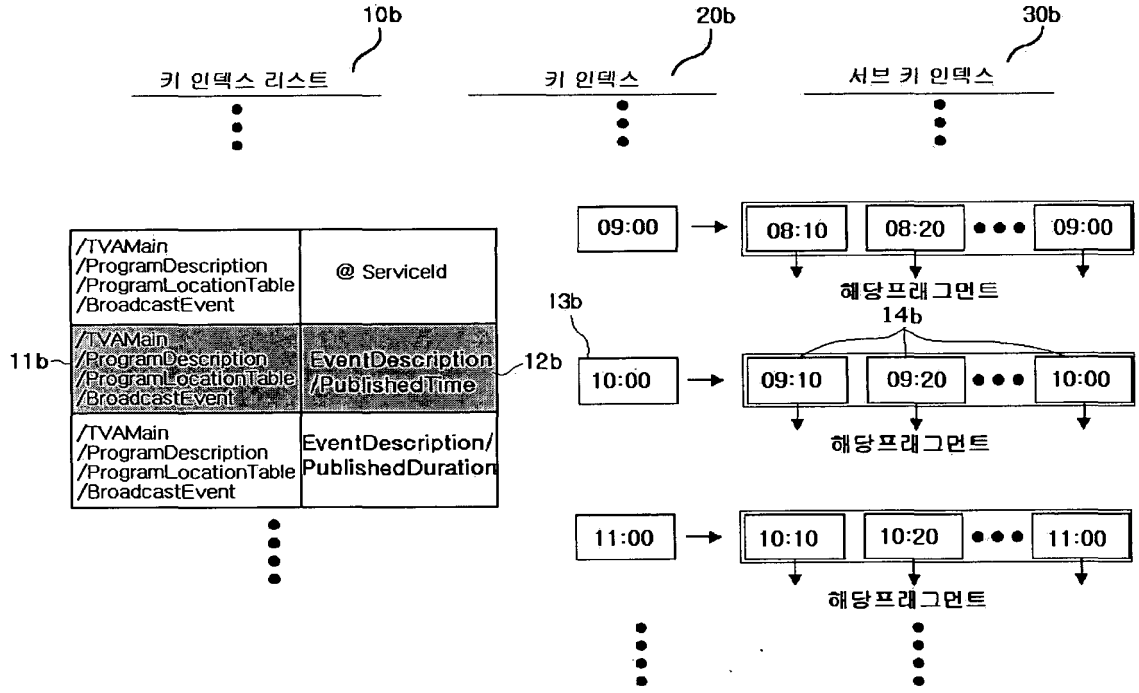
【도 6】


키 인덱스 리스트 섹션
10
fragment_xpath_ptr #1
key_xpath_ptr #1
index_container #1
key_index_identifier #1
11
12
13
14
키 인덱스 섹션
20
key_index_identifier #1
high_key_value #11
sub_index_container #11
sub_index_identifier #11
high_key_value #12
sub_index_container #12
sub_index_identifier #12
서브키 인덱스 섹션
30
sub_index_identifier #11
key_value #111
target_container #111
target_handle #111
key_value #112
target_container #112
target_handle #112
메타데이터 프래그먼트
메타데이터

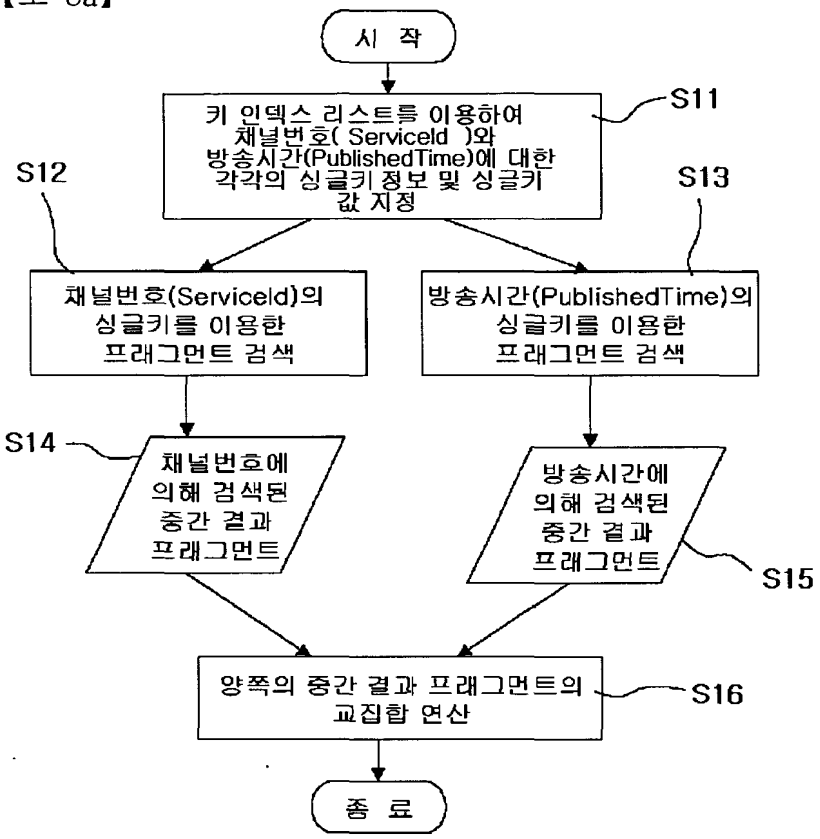
【도 7a】


10a
키 인덱스 리스트
20a
키 인덱스
30a
서브 키 인덱스
11a
12a
/TVAMain /ProgramDescription /ProgramLocationTable /BroadcastEvent
@ ServiceId
/TVAMain /ProgramDescription /ProgramLocationTable /BroadcastEvent
EventDescription /PublishedTime
/TVAMain /ProgramDescription /ProgramLocationTable /BroadcastEvent
EventDescription/ PublishedDuration
13a
509
519
529
14a
500
501
509
510
511
519
520
521
529
해당프래그먼트

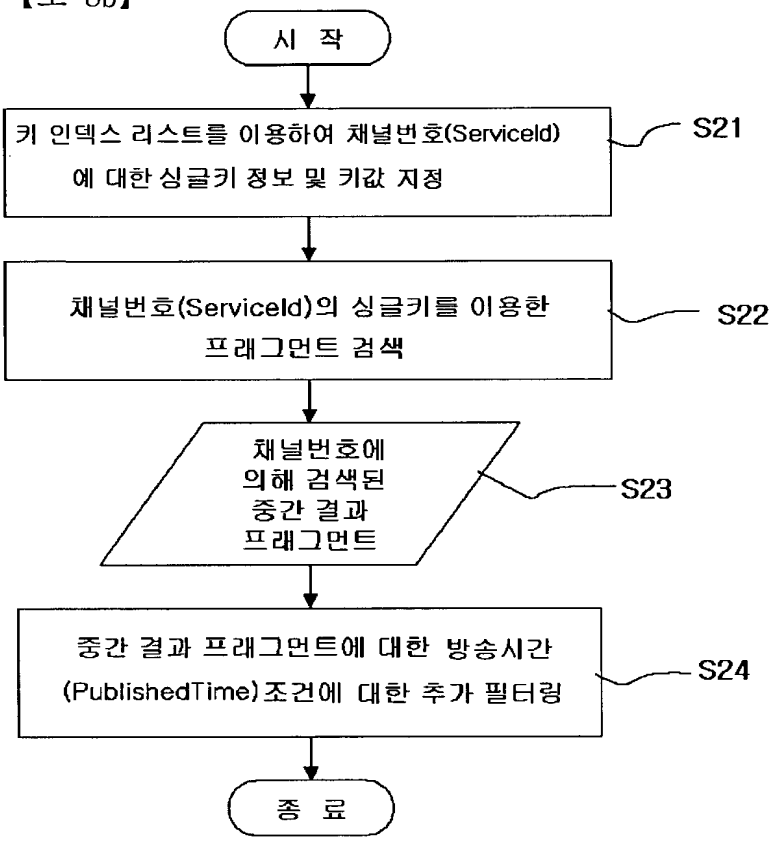
【도 7b】


10b
키 인덱스 리스트
20b
키 인덱스
30b
서브 키 인덱스
/TVAMain /ProgramDescription /ProgramLocationTable /BroadcastEvent
@ ServiceId
11b
/TVAMain /ProgramDescription /ProgramLocationTable /BroadcastEvent
EventDescription /PublishedTime
12b
/TVAMain /ProgramDescription /ProgramLocationTable /BroadcastEvent
EventDescription/ PublishedDuration
13b
09:00
10:00
11:00
14b
08:10
08:20
09:00
09:10
09:20
10:00
10:10
10:20
11:00
해당프래그먼트

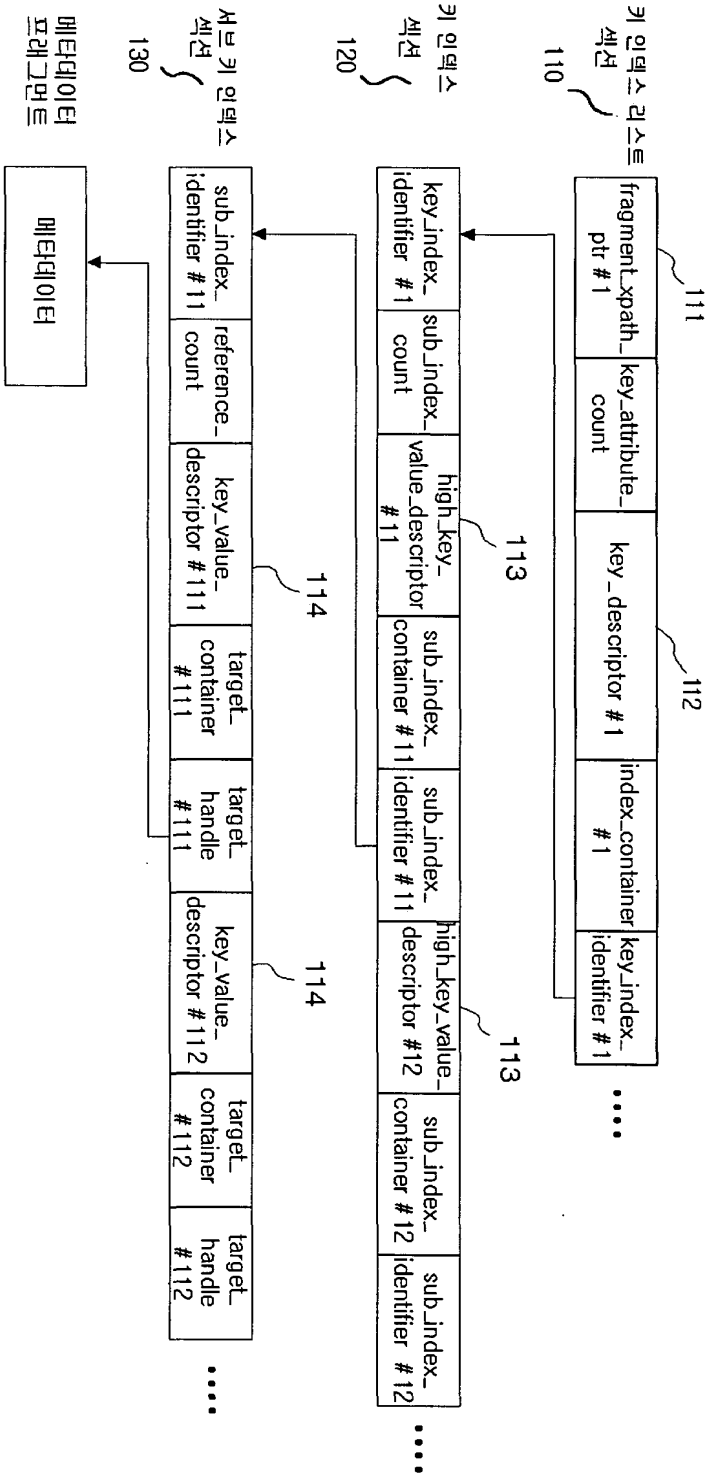
【도 8a】


시 작
키 인덱스 리스트를 이용하여 채널번호(ServiceId)와 방송시간(PublishedTime)에 대한 각각의 싱글키 정보 및 싱글키 값 지정
S11
S12
S13
채널번호(ServiceId)의 싱글키를 이용한 프래그먼트 검색
방송시간(PublishedTime)의 싱글키를 이용한 프래그먼트 검색
S14
채널번호에 의해 검색된 중간 결과 프래그먼트
방송시간에 의해 검색된 중간 결과 프래그먼트
S15
양쪽의 중간 결과 프래그먼트의 교집합 연산
S16
종 료

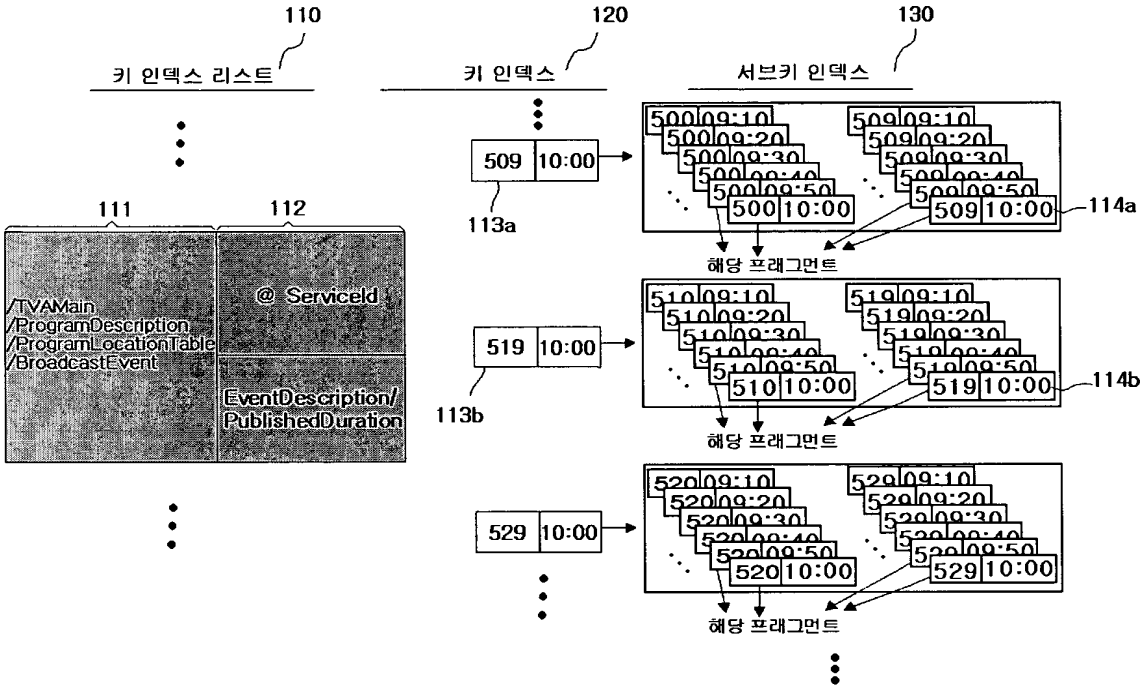
【도 8b】


시 작
키 인덱스 리스트를 이용하여 채널번호(ServiceId)
에 대한 싱글키 정보 및 키값 지정
S21
채널번호(ServiceId)의 싱글키를 이용한
프래그먼트 검색
S22
채널번호에
의해 검색된
중간 결과
프래그먼트
S23
중간 결과 프래그먼트에 대한 방송시간
(PublishedTime)조건에 대한 추가 필터링
S24
종 료

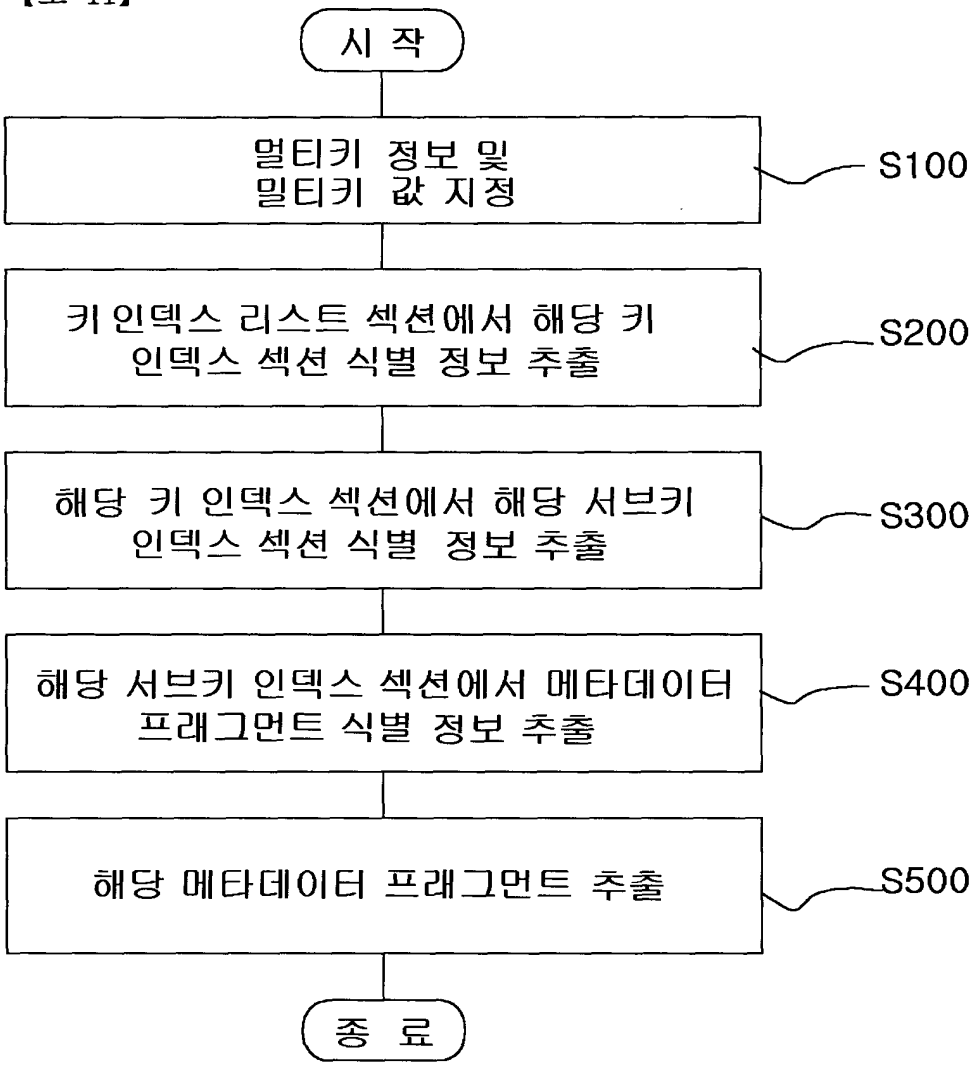
【도 9】


키 인덱스 리스트 섹션
110
fragment_xpath_ptr # 1
key_attribute_count
key_descriptor # 1
index_container # 1
key_index_identifier # 1
111
112
113
키 인덱스 섹션
120
key_index_identifier # 1
sub_index_count
high_key_value_descriptor # 11
sub_index_container # 11
sub_index_identifier # 11
high_key_value_descriptor #12
sub_index_container # 12
sub_index_identifier # 12
114
서브 키 인덱스 섹션
130
sub_index_identifier # 11
reference_count
key_value_descriptor # 111
target_container #111
target_handle #111
key_value_descriptor # 112
target_container #112
target_handle #112
메타데이터 프래그먼트
메타데이터

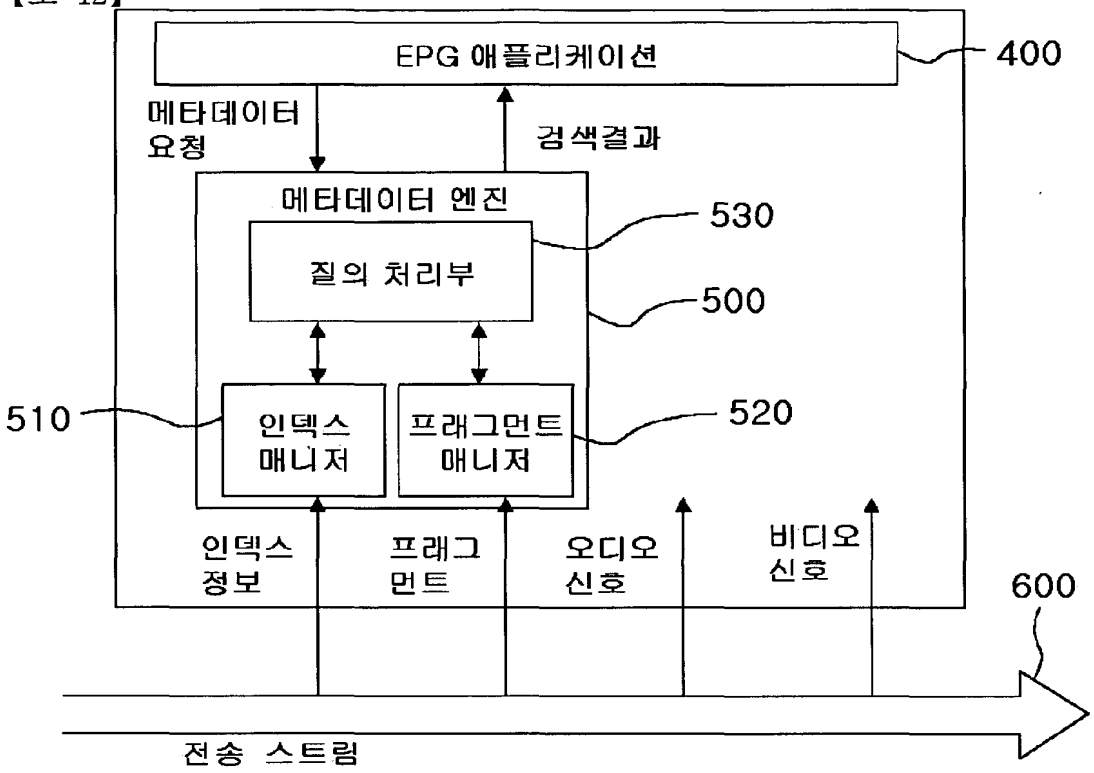
【도 10】


110
키 인덱스 리스트
120
키 인덱스
130
서브키 인덱스
111
112
/TVAMain
/ProgramDescription
/ProgramLocationTable
/BroadcastEvent
@ ServiceId
EventDescription/
PublishedDuration
509 10:00
113a
519 10:00
113b
529 10:00
500 10:00
509 10:00
114a
510 10:00
519 10:00
114b
520 10:00
529 10:00
해당 프래그먼트
해당 프래그먼트
해당 프래그먼트

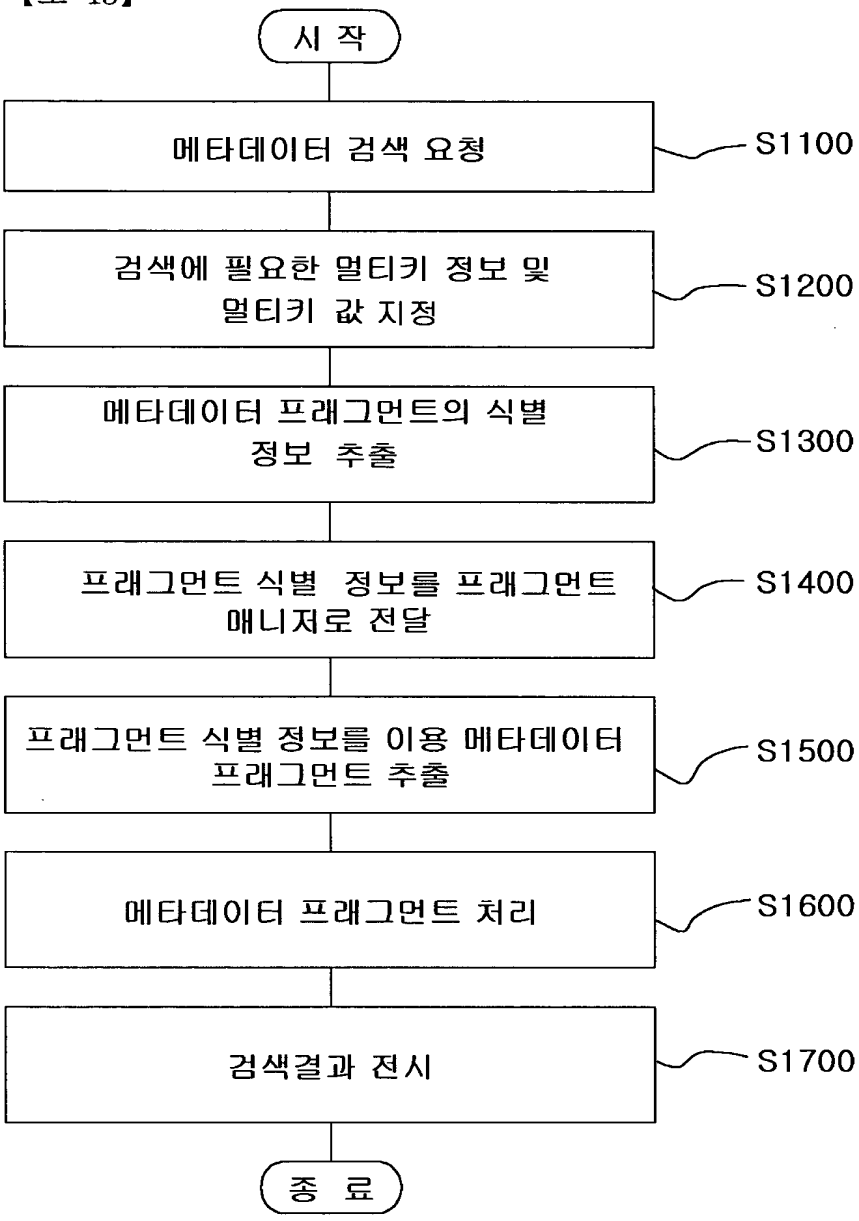
【도 11】


시 작
멀티키 정보 및
밀티키 값 지정
S100
키 인덱스 리스트 섹션에서 해당 키
인덱스 섹션 식별 정보 추출
S200
해당 키 인덱스 섹션에서 해당 서브키
인덱스 섹션 식별 정보 추출
S300
해당 서브키 인덱스 섹션에서 메타데이터
프래그먼트 식별 정보 추출
S400
해당 메타데이터 프래그먼트 추출
S500
종 료

【도 12】


EPG 애플리케이션
400
메타데이터 요청
검색결과
메타데이터 엔진
530
질의 처리부
500
510
인덱스 매니저
프래그먼트 매니저
520
인덱스 정보
프래그먼트
오디오 신호
비디오 신호
600
전송 스트림

【도 13】


시 작
메타데이터 검색 요청
S1100
검색에 필요한 멀티키 정보 및 멀티키 값 지정
S1200
메타데이터 프래그먼트의 식별 정보 추출
S1300
프래그먼트 식별 정보를 프래그먼트 매니저로 전달
S1400
프래그먼트 식별 정보를 이용 메타데이터 프래그먼트 추출
S1500
메타데이터 프래그먼트 처리
S1600
검색결과 전시
S1700
종 료